UNIVERSIDAD AUTONOMA DE MADRID

ESCUELA POLITECNICA SUPERIOR

Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación

TRABAJO FIN DE GRADO

Tapones sincronizados con medida de tiempo

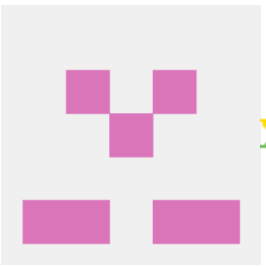
Pedro Granados López
Tutor: Kostadin Nedeltchev Koroutchev

JUNIO 2021

Tapones sincronizados con medida de tiempo

AUTOR: Pedro Granados López
TUTOR: Kostadin Nedeltchev Koroutchev

Grupo de Aprendizaje Automático (GAA)
Dpto. de Ingeniería Informática
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Junio de 2021



Resumen (castellano)

Este Trabajo de Fin de Grado consiste en ofrecer un dispositivo capaz de llevar un control de tiempo, que nos permita sin necesidad de la ayuda de otras personas saber el momento en el que se tenga que realizar una toma de cualquier medicamento o suplemento especial. Aunque puede ser para cualquier usuario de todas las edades, la elaboración de este proyecto está encaminada a personas que tengan que depender de recordatorios de otras personas como personas mayores o con alguna deficiencia.

Hoy en día estamos en una situación difícil debido a la pandemia originada por la COVID-19. Esta situación ha provocado que las relaciones entre personas se hayan visto afectadas de manera drástica, hasta el punto de que aquellas personas que necesitan de la ayuda de otras no han podido recibir en muchos casos la atención necesaria que requerían estas situaciones. La idea de este trabajo tiene como uno de sus principales objetivos, ofrecer una nueva opción y ayuda para aquellas personas que necesiten realizar la toma de medicamentos sin necesidad de depender del recordatorio de otras personas.

En líneas generales el dispositivo que se propone en este Trabajo de Fin de Grado puede ser capaz de adherirse a una tapa de un bote de medicamentos o pastillero, el cual se compone de una pantalla que sea capaz de representar un contador progresivo con la fecha actualizada gracias a un smartphone Android que pasará esta información por medio de una conexión bluetooth a un módulo bluetooth configurado previamente en nuestro dispositivo. Tendrá una fuente de alimentación portátil recargable y la funcionalidad global se conseguirá gracias a una placa de desarrollo de hardware de Arduino. Se han desarrollado dos prototipos, los cuales se han conseguido sincronizar desde un mismo dispositivo móvil. La sincronización y conexión con ambas tapas se realizará desde una aplicación Android desarrollada para estos objetivos.

Palabras clave (castellano)

Módulo bluetooth HC-06, display OLED, contador progresivo, Arduino UNO, placa de desarrollo hardware, módulo TTL, comandos AT, bloques de funcionalidad, interfaz, layout.

Abstract (English)

This Bachelor Thesis consists of offering a device capable of take control of the time, which allows us without the help of other people to know the exact time at which we have to take any medication or special supplement. Although it can be for any user of all ages, the development of this project is aimed at people who have to rely on reminders from others such as the elderly or people with some deficiency.

Nowadays we are in a difficult situation due to the pandemic caused by COVID-19. This situation has changed the relationships between people in a drastic way, to the point that those people who need the help of others have not been able to get in many cases the necessary attention that their situations required. The idea of this work has as one of its main objectives, to offer a new option and help for those people who need to take medication without having to rely on the reminder of other people.

In general terms, the device proposed in this Bachelor Thesis can be able to be attached to a cap of a medicine bottle or pillbox, which is composed of a display that is able to represent a progressive counter with the updated date thanks to an Android smartphone that will pass this information through a bluetooth connection to a bluetooth module previously configured in our device. It will have a portable rechargeable power supply and the overall functionality will be achieved thanks to an Arduino hardware development board. Two prototypes have been developed, which have managed to synchronize from the same mobile device. The synchronization and connection with both caps will be done from an Android application developed for these purposes.

Keywords (inglés)

HC-06 bluetooth module, OLED display, progressive counter, Arduino UNO, hardware development board, TTL module, AT commands, functionality blocks, interface, layout.

Agradecimientos

Una vez finalizado este Trabajo Fin de Grado con todo el esfuerzo que ha conllevado su elaboración, llega otro momento en el que aunque no se requieren conocimientos técnicos adquiridos a lo largo de estos años no se hace más fácil de expresar, ni mucho menos de elegir las palabras adecuadas. Seguramente sea la parte más sentimental (moñas) pero después de tantos años de duro esfuerzo, me apetece explayarme y mostrarme más como estoy de agradecido a todas aquellas personas que aparecerán en estos agradecimientos.

En primer lugar, me gustaría agradecer a mi tutor de este Trabajo de Fin de Grado, Kostadin. Gracias por ayudarme y ofrecerme la oportunidad de trabajar contigo en un proyecto en el cual me ha encantado aprender y elaborar un prototipo creado que era justo lo que andaba buscando cuando empecé a hablar con posibles tutores. No ha sido una época fácil, ha habido muchas trabas por el camino, pero al final hemos conseguido en líneas generales lo que buscábamos. Gracias por tu ayuda y confianza en mí.

Gracias a mis amigos del grado que por medio de la unión de conocimientos (y penas) hemos conseguido (Unos antes, otros ahora y otros llegarán muy pronto) llegar a la meta que nos propusimos desde el primer día en el que entramos en la EPS. Seguro que el futuro que tenemos por delante es apasionante y tenemos que aprovecharlo para dar lo mejor de nosotros mismos.

Me gustaría también agradecer a todos y cada uno de los profesores que he tenido a lo largo de mi vida académica (de verdad, aunque tuviera mis más y mis menos con alguno en mi foro interno). Gracias por las enseñanzas de todos ellos y en especial agradecer a aquellos con los que he podido tener una relación más estrecha que me han aportado una confianza que seguramente sin ella no estaría escribiendo este trabajo.

AVISO IMPORTANTE: A partir de aquí nos adentramos en el momento más emotivo en el que seguramente (aunque no sea mucho de llorar) se me pueda caer alguna lágrima recordando todo el camino.

Aunque solo haya conocido a mi abuela materna, siempre me ha encantado saber todo lo posible sobre cómo eran (y es en el caso de mi abuela) mis abuelos y abuela paterna. Aunque no haya tenido la suerte de escuchar sus historias de sus bocas, he conseguido ver en una medida seguramente muy pequeña como eran y siempre he intentado ser y actuar como seguramente les hubiera gustado verme. Les estoy muy agradecidos a mis abuelos y abuelas porque siempre me ha gustado pensar en ellos porque han sido una gran motivación para intentar ser mejor y que si de alguna manera se pudieran sentir orgullosos de mí, así fuese.

Gracias a mis amigos de sanse que de verdad han sido verdaderos amigos que, aunque se cuenten con los dedos justos me han ayudado en muchos momentos muy difíciles a remar contra viento y marea y seguramente sin su ayuda tampoco estaría aquí.

Me gustaría realizar una mención muy especial. Gracias infinitas a mi profesora de 3º de Primaria María, porque después de los dos primeros años de Primaria en los que el tutor que tenía no daba nada por mi (le dijo a mis padres que no llegaría a nada en la vida), me diste una confianza tal, que estoy seguro que sin tu ayuda en un momento tan importante en mi vida no hubiera llegado ni la mitad de lejos. Me acuerdo de ti absolutamente todos los días y espero que allá donde estés, estés orgullosa de mí y ojalá pudiese habértelo dicho en persona, pero te estaré eternamente agradecido.

Aunque solo lleves 9 meses conmigo después de haberte esperado durante 24 años, doy gracias a mi husky Luke porque hemos creado un vínculo que me ha dado una fuerza y confianza tremenda que jamás pensé que podría tener.

Muchas gracias a mi hermano que, aunque piense que no va a ir en este escrito le toca también una mención especial. Tenemos siempre mejores y peores momentos como todos los hermanos, pero también me has ayudado mucho en este camino y esos piques sanos han sido muy buenos para ambos en nuestros caminos. Te reto a que dentro de dos-tres años, cuando tengas la posibilidad de escribir tu TFG tengas una mención sobre mí mas buena que la que hago sobre ti. Muchas gracias, Marcos.

Por último y no menos importante, gracias infinitas a mis padres. A parte de lo evidente que es darme una vida, sois los mejores padres que jamás podría haber tenido. Lo habéis dado todo por mí, habéis soportado mis malos momentos con buenas caras y dándome ánimos y empujones que han sido determinantes en mi vida. Siempre que me he caído ahí estabais vosotros para levantarme y ayudarme a llegar hasta donde estoy ahora escribiendo esto con el corazón. Nunca podré devolveros todo lo que me habéis dado, pero haré lo posible en el futuro porque estéis la mitad de orgullosos de mi como yo lo estoy de vosotros que es infinito. Gracias de verdad porque (junto con la saga de Star Wars que tenía que aparecer si o sí) me habéis criado de una manera en la que me considero con la fuerza e independencia suficiente para afrontar mi futuro con ilusión y determinación.

Finalmente, agradecer a cualquier persona que haya leído este trabajo (y en caso de los agradecimientos, esta chapa) y espero que haya encontrado en este trabajo aquello que buscaba.

INDICE DE CONTENIDOS

INDICE DE FIGURAS

INDICE DE TABLAS

1 Introducción

1.1 Motivación

Esta memoria de Trabajo de Fin de Grado (a lo largo de esta memoria también se llamará como TFG) trata sobre la elaboración de un prototipo de tapas para pastilleros o botes de medicamentos, los cuales por medio de una conexión *bluetooth* se conectarán a un dispositivo móvil con sistema operativo Android que nos avisará de que hay que realizar una toma de lo que se encuentre en el interior del bote de la tapa.

Hoy en día el COVID-19 ha cambiado por completo nuestra manera de ver el mundo y de comportarnos. En lo que se refiere a nuestra salud, ha sido un cambio importante, ya que nos preocupamos en general más por nuestra protección frente a cualquier enfermedad. Al inicio de esta pandemia, este sentimiento de protección fue una necesidad sobre todo con nuestros mayores, pero en la actualidad se ha convertido en algo rutinario y por ello la preocupación del cuidado de personas que tienen dificultades de valerse por sí mismas se ha visto aumentada.

Esta memoria ha sido una gran motivación para mí, ya que en un inicio se propuso esta idea, como unas tapas de envases de medicamentos, las cuales permitan a personas mayores o con problemas de dependencia, tener una ayuda adicional para que puedan realizar la toma de sus medicamentos diarios sin que tengan la necesidad de que otras personas les recuerden en determinados momentos las horas de estas tomas.

Pero no solo resulta una ayuda para aquellas personas que necesiten de recordatorios de sus tomas, sino que para cualquier persona que tenga una enfermedad puntual que necesite de la toma de ciertos medicamentos, supone también una ayuda por la cual no necesitará estar pendiente de una hora ya que la alarma previamente establecida, se encargará de realizar ese trabajo. Para el caso de los niños también puede ser una buena opción, en el caso de que se encuentren solos en casa en determinados momentos en que puedan estar enfermos y necesiten realizar tomas de determinados medicamentos o vitaminas especiales.

1.2 Objetivos

Para la realización de este proyecto, se han establecido una serie de objetivos los cuales pretenden diseñar y construir un prototipo funcional para cualquier tapa ya sea de pastillero o de bote individual. Para ello se han hecho uso de algunos programas de diseño de dispositivos y de programación para los entornos de Arduino y Android.

En el proceso de alcanzar este objetivo se han seguido los siguientes puntos que se irán explicando en detalle a lo largo de esta memoria de TFG:

- Primero se ha procedido a estudiar e investigar qué tipo de tapas con bote o pastilleros existen con funcionalidades parecidas, de los cuales se pretende obtener referencias para poder realizar un diseño, acorde a lo que está estipulado en el mercado.

- Otro objetivo, ha sido el del estudio previo de los componentes *hardware* capaces de ofrecer aquellas funcionalidades necesarias para el prototipo y que podían desarrollarse por medio del *software* de Arduino
- Para la conexión deseada entre *smartphone* y prototipo de la tapa, se ha puesto como objetivo la realización de una aplicación móvil. Para la elaboración de la aplicación en Android, se han estudiado distintos entornos de programación, con el propósito de ver y elegir cual ofrece los elementos necesarios para el desarrollo del proyecto.
- Por medio de la aplicación desarrollada en Android y el *software* de Arduino programado, se ha establecido una conexión entre un dispositivo Android y el *hardware* del prototipo, con el objetivo de transferir la información deseada a las tapas.
- Una vez conseguidos los objetivos globales propuestos, se han llevado a cabo objetivos secundarios en los que se ha tratado de mejorar en la medida de lo posible ciertos elementos correspondientes al prototipo o a la aplicación móvil.
- También se ha documentado un manual de usuario que sirva de ayuda para entender el manejo de la aplicación junto con el prototipo desarrollado.

1.3 Organización de la memoria

Esta memoria de Trabajo de Fin de Grado se organiza de la siguiente manera:

- Punto 1: Introducción. En este primer punto introduce cuales han sido mis motivaciones, los objetivos que se han perseguido y como se distribuye la memoria en la realización de este TFG.
- Punto 2: Estado del arte. En el punto 2 se encuentra el estado del arte en el que se dará una visión global sobre las distintas opciones de mercado que cumplen una función similar a lo que se propone en este Trabajo de Fin de Grado. Una vez estudiadas las diferentes opciones, se expondrán y explicarán los motivos por los que se han utilizado las distintas herramientas con las que se ha trabajado.
- Punto 3: Diseño y desarrollo del prototipo. En este punto, se han detallado las distintas etapas en el desarrollo y diseño del prototipo.
- Punto 4: Pruebas realizadas. En este punto se expondrán de manera breve y sin entrar en muchos detalles que tipo de pruebas se han ido realizando en cada etapa.
- Punto 5: Conclusión y desarrollos futuros. En el último punto de este Trabajo Fin de Grado se exponen una serie de conclusiones a las que se han llegado en la realización de este proyecto. Para finalizar, se han pensado varias mejoras a desarrollar y objetivos a alcanzar en el futuro que pretenden dar mejores funcionalidades y optimizaciones tanto de espacio como tecnológicas.

2 Estado del arte

2.1 Dispositivos en el mercado con funciones similares

En el momento de la realización de este TFG, existen en el mercado opciones que guardan alguna similitud en cuanto al funcionamiento y tipo de uso, con lo que se pretende conseguir en este trabajo.

Para el caso de los botes más pequeños, el mecanismo de sus tapas es el más sencillo ya que se componen, en su mayoría, de un contador digital que utiliza como fuente de alimentación una pila y tiene como particularidad que el tiempo de conteo finaliza al abrir la tapa y una vez se cierra el bote con la tapa vuelve a contar reiniciando el tiempo hasta la siguiente toma. El formato del contador de este tipo de tapas representa las horas y minutos transcurridos. Los ejemplos de la figura 2-1 pertenecen a la empresa TimerCap.

Figura 2-1: Ejemplos de tapas con contador convencional para botes individuales. (*extraído de* https://www.timercap.com/product-page/standard-4-pack)

Hay otros casos como el anterior en el que las tapas son también de botes individuales, que indican la hora y minutos en los que se realizó la última toma y se pueden establecer varias alarmas para las siguientes tomas indicando también los días en los que se tendrían que realizar estas tomas. Tiene un uso distinto al anterior ya que aquí no hay un contador que lleve el tiempo, sino que se indica solo la hora de la última toma y a parte tiene la alarma en función de la hora establecida para la siguiente toma como elemento distintivo de la tapa anterior. El ejemplo de la figura 2-2 pertenece a la empresa E-pill.

Figura 2-2: Ejemplo de tapa con indicador de última toma y alarma para botes individuales. (*extraído de* https://www.epill.com/timecap.html)

Por otro lado, se encuentran pastilleros (el ejemplo en la figura 2-3, perteneciente a la empresa Tabtime) que permiten tener más medicaciones diferentes, que en la gran mayoría se pueden distribuir en tres tomas semanales, aunque hay casos más específicos que establecen tomas genéricas por días o incluso en menos días que los semanales. Los pastilleros de este tipo suelen tener una alarma con sonido que permiten establecer en la gran mayoría de ocasiones más de una alarma, para controlar más tomas diarias y suelen incluir un altavoz de sonido regulable y en determinados casos un zumbador que hace vibrar al dispositivo.

En algunos casos más especiales existen pastilleros que aparte de las funcionalidades que se acaban de exponer, permiten coger los medicamentos gracias a la apertura de un compartimento que indica la toma correspondiente a la programada por la alarma anteriormente establecida, sin necesidad de abrir completamente la tapa.

Figura 2-3: Ejemplo de pastillero con contador convencional y alarma. (*extraído de* https://es.tabtime.com/collections/automatic-pill-dispensers/products/automatic-pill-dispenser)

Estos dispositivos intentan ofrecer una ayuda extra para la toma de medicamentos, en la gran mayoría de los casos enfocados para personas mayores que puedan tener problemas de independencia y no tengan a alguien cercano que pueda ayudarles a seguir con la medicación de manera correcta.

Como cualquier elemento que se pueda encontrar en el mercado, este tipo de productos ofrecen una serie de ventajas y desventajas que toda persona que quiera adquirirlos deberá estudiar y sopesar para saber cuál se adapta mejor a sus necesidades.

Por un lado está el precio, que dependiendo de las funcionalidades que ofrezcan y los elementos que incluyan, será más alto o más bajo. Para el caso más sencillo inicialmente descrito de la tapa con contador simple el precio puede rondar los 12 euros por tapa. Si nos vamos a la tapa del segundo caso el precio aumenta hasta los 45 euros estableciendo alguna funcionalidad más como la alarma e indicador diario. En el caso de los pastilleros, hay más tipos y el precio varía más, pero el precio mínimo puede rondar los 20 euros en los que tienen funcionalidades más simples, pero en la gran mayoría de casos tienen precios bastante más elevados pudiendo llegar hasta los 150 euros.

En lo que corresponde al uso, el primer caso es el más simple. Su uso es muy básico ya que lo único que hay que hacer es tapar el bote con la tapa y empezará el contador a funcionar.

En el segundo ejemplo, la tapa tiene un uso algo menos intuitivo ya que establece el tiempo de la última toma y para programar la alarma en función del día hay que realizar una serie de pasos que vienen especificados en su manual de usuario, con los botones que vienen incorporados en su tapa. Para los pastilleros el uso es muy parecido al segundo, en su mayoría, pero al incluir botones con funcionalidades más claras y tener tamaños mayores, hace que el uso sea algo más sencillo e intuitivo en el caso de los pastilleros.

En cuanto a autonomía y movilidad es claro que el caso más favorable y que más ventajas tiene es el primer tipo de tapa ya que solo se compone del contador y no requiere de más funcionalidades lo que le da una autonomía de aproximadamente un año, su batería puede ser intercambiable y en el caso de la movilidad al ser botes individuales, al igual que en el segundo caso, son los que mejores ventajas ofrecen en este apartado, debido a sus tamaños. La autonomía para las tapas del segundo caso es de aproximadamente la mitad de las anteriores con una vida útil de 6 meses. La batería de los pastilleros suele ser en su mayoría de 4 pilas AAA que ofrecen una autonomía menor a las dos opciones anteriores y aunque también pueden transportarse bien, son más grandes que los casos anteriores.

Una vez establecida una visión general de los tipos de tapas con sistemas de control de tiempo que existen actualmente en el mercado y como se ha indicado en los objetivos, este Trabajo de Fin de Grado pretende aportar otra idea para dar más opciones de mercado, intentando aportar un producto de uso sencillo aprovechándose de que hoy en día está muy extendido también a los más mayores el manejo de los *smartphones*. En el siguiente apartado del estado del arte, se expondrán y explicarán que herramientas y entornos se han utilizado, para desarrollar y diseñar un sistema que pueda ofrecer funcionalidades y diseños válidos y competentes a los ya existentes en los modelos vistos.

2.2 Entornos y herramientas utilizadas

En este punto se mostrará una visión global de los programas y herramientas utilizadas, mostrando los distintos entornos en los que se han trabajado y que han sido claves para el desarrollo del prototipo inicial. Los entornos utilizados han sido Arduino, Fritzing, *App Inventor* y un soldador con estaño para la unión de los distintos componentes.

2.2.1 Arduino

En primer lugar, he decidido hacer esta parte del proyecto en Arduino debido a su gran capacidad para desarrollar proyectos *software* y *hardware* y a que ya tengo experiencias previas tanto en otros proyectos personales como en proyectos realizados en algunas asignaturas que he cursado en el grado. Esta versatilidad unida a la gran variedad de componentes que están en el mercado que permiten su programación con el *software* de Arduino también han sido claves en la elección. Arduino es uno de los entornos de desarrollo libre de *software* y *hardware* más extendido a nivel mundial. Permite gracias a sus placas de desarrollo la posibilidad de configurar y construir dispositivos digitales para una infinidad de funcionalidades.

Para este proyecto en particular distinguiremos por un lado la parte de *software* y por otro la parte de *hardware*. La parte de *software* se basa en el código programado para que, por medio de la llamada a los distintos pines de la placa de desarrollo de Arduino, podamos pasar su funcionalidad a los componentes *hardware* que componen el proyecto. Por este

motivo, se pasará primero a mostrar la parte correspondiente a la programación de los componentes y seguidamente la parte respectiva a los componentes utilizados.

Software

Aquí se hablará en líneas generales del código programado para este proyecto y lo que se encarga de hacer en cada caso. Este código se mostrará y explicará en más detalle en el punto 3 de desarrollo y diseño.

En la parte de software de Arduino se ha programado un código con funciones del lenguaje de programación C y el lenguaje *Processing* (basado en Java) los cuales se componen de las instrucciones que necesitaremos para establecer las funcionalidades del prototipo. El objetivo de este código se centra en establecer una interfaz para la pantalla la cual mostrará un contador progresivo en el siguiente formato "horas:minutos:segundos". Debajo de este contador se establece la fecha con formato en "día:mes:año".

Por otro lado, se ha programado la parte de la conexión con el módulo *bluetooth* para la recepción y la transferencia de información entre *smartphone* y pantalla. Para los módulos *bluetooth*, también se han empleado comandos de configuración previos para establecer distintos nombres para que no surjan conflictos en el *smartphone* a la hora de establecer conexión con varias tapas a la vez.

Para que este código sea funcional me he basado en el funcionamiento de las placas de desarrollo de *hardware* y de los componentes que se han utilizado. Aunque se explicarán más en detalle en la parte de hardware y de diseño.

Hardware

Este punto se basa en el entorno de los elementos físicos que van a componer el proyecto en este caso serán dispositivos digitales y de control. El elemento central del control del prototipo es una placa de desarrollo de *hardware* (ejemplo utilizado en la figura 2-4), que será la encargada de, en el inicio de realizar la conexión PC-placa para poder subir la parte de código programada al controlador que según lo indicado en el programa distribuirá al resto de componentes. La alimentación y conexión se realizará inicialmente por medio de USB. El control de cada dispositivo que queramos que intervenga en nuestro proyecto, tendrá que conectarse al pin de entrada que corresponda de la placa, que normalmente los llamaremos a través de nuestro código a no ser que estén definidos por defecto.

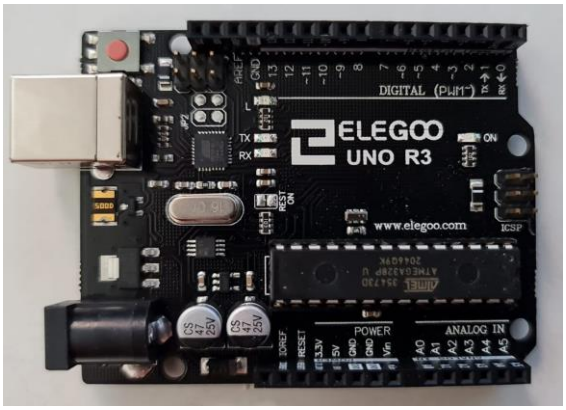


Figura 2-4: Ejemplo de placa de desarrollo Arduino utilizada.

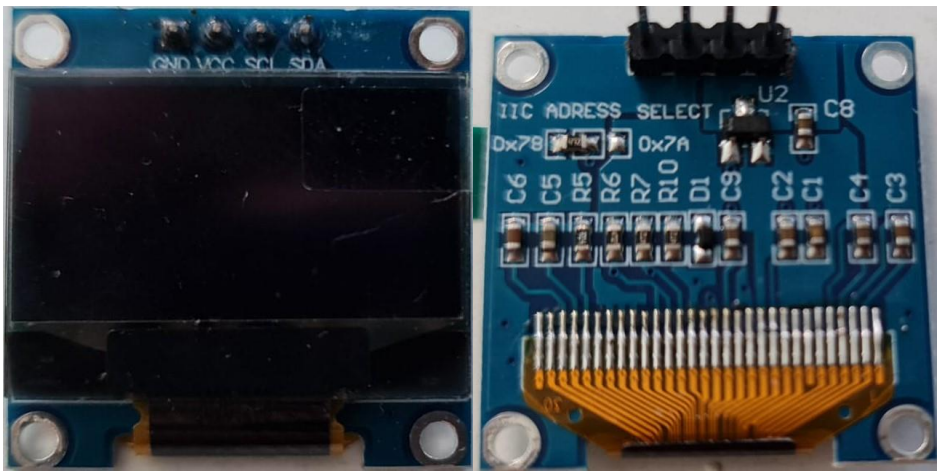
Para la conexión inicial entre los distintos componentes digitales de Arduino que componen el prototipo y por una mayor comodidad a la hora de cambiar o aplicar nuevas ideas, se ha empleado una placa *protoboard*. A diferencia de la soldadura que se ha empleado como parte final de unión de componentes, esta placa ha permitido poder probar componentes sin necesidad de conectarlos de manera definitiva.

Los módulos *bluetooth* utilizados para este proyecto son los llamados HC-06 (ejemplo utilizado en la figura 2-5), cuya principal funcionalidad es la de aportar una conexión a un *smartphone* o cualquier otro elemento que disponga de conexión *bluetooth* y sistema operativo de Android (No aceptan conexiones con el sistema operativo IOS) a proyectos Arduino. Se realiza una configuración previa en la parte de *software* mediante los comandos AT, en la que podemos modificar algunos elementos como el nombre, la velocidad o el código PIN de conexión previa al dispositivo. Este dispositivo en concreto trabaja siempre en modo esclavo, que se encargará para recibir peticiones de conexión.

(a) Vista superior. (b) Vista inferior.

Figura 2-5: Ejemplo de módulo *bluetooth* HC-06 Arduino utilizado.

Para este TFG las pantallas que se han utilizado han sido *displays* OLED de 0.96” (ejemplo utilizado en la figura 2-6). Estas pantallas consiguen una resolución de 128*64 *pixels*, los cuales se pueden configurar individualmente. Se ha decidido que estas pantallas sean de tipo OLED porque tienen una buena iluminación, por lo que no necesitan de retroiluminación, como si la necesitan las pantallas LCD. Tienen un consumo bajo que también lo hace interesante si tenemos en cuenta que es un prototipo que dependerá constantemente de una batería individual.

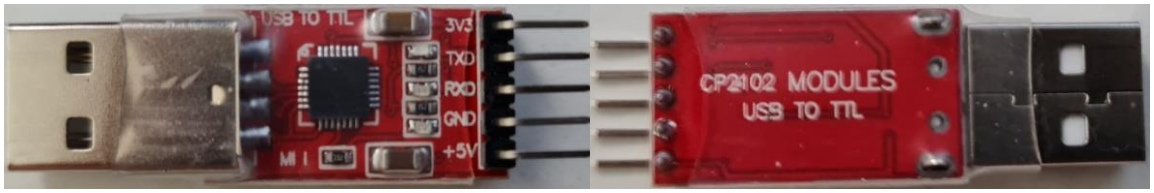


(a) Vista superior. (b) Vista inferior.

Figura 2-6: Ejemplo de *display* OLED utilizado.

Para la alimentación del dispositivo se han usado *powerbanks* de 10000mah y otra más pequeña pensada para prototipos más pequeños de 5000mah que ambas funcionan a 5V. Gracias a estas fuentes de alimentación y una vez subido el código a la placa de desarrollo, podremos crear un prototipo sin necesidad de tener que estar conectados por medio de USB a un PC. Tener capacidades tan altas en estas fuentes de alimentación nos permite que tengamos autonomías de baterías en ambos casos aceptables. Se han tenido en cuenta varias opciones a la hora de alimentar al prototipo, como pilas recargables u otro tipo de baterías, pero en relación con el rendimiento, autonomía del proyecto y que estaban en mi posesión previamente a la realización de este proyecto en primera instancia se ha decidido usar este tipo de alimentación.

Para la fuente de alimentación se ha utilizado un módulo TTL (ejemplo utilizado en la figura 2-7), el cual también facilita comunicaciones entre placas de desarrollo de Arduino y PC. Por medio de su conexión USB, nos servirá de conexión entre fuente de alimentación y placa de desarrollo que gracias a que permite 5V podremos alimentar sin problemas de voltaje todos los elementos de Arduino con la *powebank*.



(a) Vista superior. (b) Vista inferior.

Figura 2-7: Ejemplo de módulo TTL utilizado.

2.2.2 Fritzing

Esta herramienta es una licencia de *software open-source* que adquirí en 2018 para ayudarme en la elaboración teórica previa de un proyecto. Su creación se basa en los principios del lenguaje de programación *Processing* y Arduino. Permite diseñar prototipos basados en Arduino y esquemas de circuitos impresos antes de pasar al apartado físico. Es una herramienta de diseño *hardware* que ha permitido realizar una visualización de las conexiones de componentes en PC como paso previo mostrar el trabajo con los componentes reales. Fritzing posee la gran mayoría de elementos con los que se pueden trabajar y configurar en Arduino.

Permite varias vistas lo que aporta mucha más información y personalización a la hora de diseñar el circuito. La parte de diseño "*Protoboard*" permite la visualización de los componentes como se ven en la vida real, que se hace mucho más intuitivo a la hora de elegir que componentes usar y como conectarlos. La pestaña de visión "Esquemático" da una visión de los componentes por bloques, donde se ven de manera más clara los elementos y pines de cada componente. La pestaña de "PCB" permite ver varias capas en caso de tratarse de casos como circuitos integrados y su enrutado entre los pines. Por último, tiene una pestaña de "Código" en la que podremos subir el código de Arduino que deseemos para la placa que tengamos conectada en ese momento.

Para este caso en particular se ha hecho más hincapié en el diseño "*Protoboard*" en el que se han dispuesto los elementos anteriormente nombrados en el apartado de *Hardware* de

Arduino. Para la fuente de alimentación al no ofrecer la posibilidad del uso de una *powerbank*, se ha utilizado una batería de litio de voltaje similar para el diseño en esta herramienta.

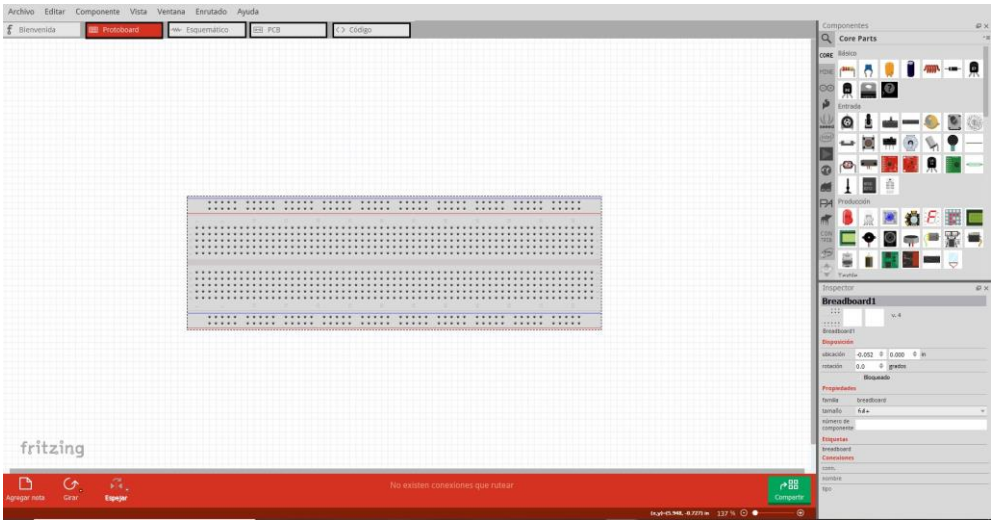


Figura 2-8: Vista de la interfaz de la herramienta Fritzing.

2.2.3 Aplicación Android

Para la realización de la aplicación se han visto y estudiado dos opciones de entornos de desarrollo integrado (IDE) de aplicaciones en el Sistema Operativo (OS) Android, elegido por su mayor uso global y disponibilidad, aparte de la comodidad personal a la hora de realizar las pruebas necesarias para este TFG con mi *smartphone* personal. Estas dos opciones son Android *Studio* y *App Inventor*. A continuación, se describirá brevemente cada una y porque se ha elegido usar para este proyecto *App Inventor*.

Android *Studio* es el programa oficial de desarrollo software para aplicaciones del OS Android sustituyendo así a su predecesor, el programa de desarrollo integrado Eclipse anunciado por Google en 2013. Está basado en el entorno de desarrollo de programas informáticos del desarrollador JetBrains llamado IntelliJ IDEA.

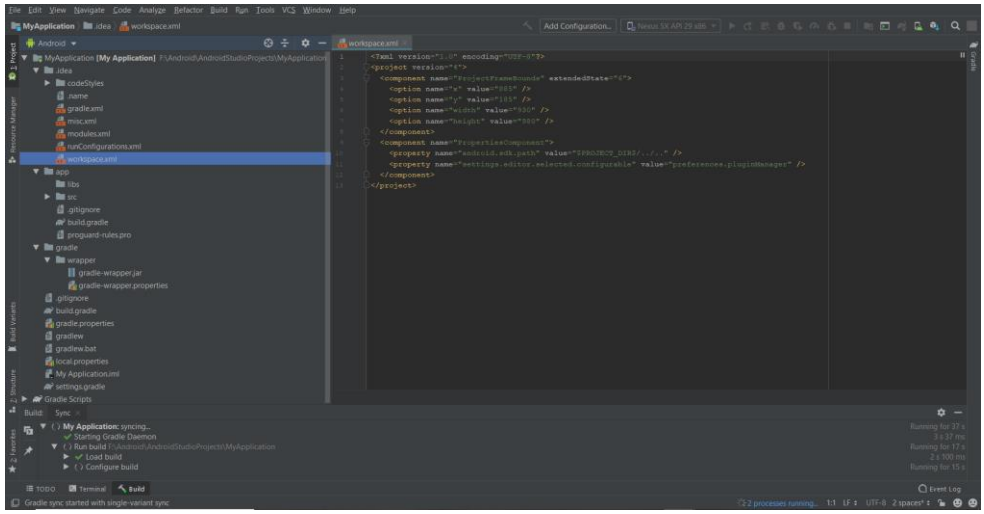


Figura 2-9: Vista de la interfaz del entorno Android *Studio*.

La segunda opción es el IDE *App Inventor* desarrollado por el laboratorio perteneciente a la Escuela de Arquitectura y Planificación del Instituto de Tecnología de Massachusetts (MIT*, Massachusetts Institute of Technology*) conocido como el *Media Lab* creado en 1985 que se dedica a la investigación en unión de los campos de la tecnología, multimedia y diseño.

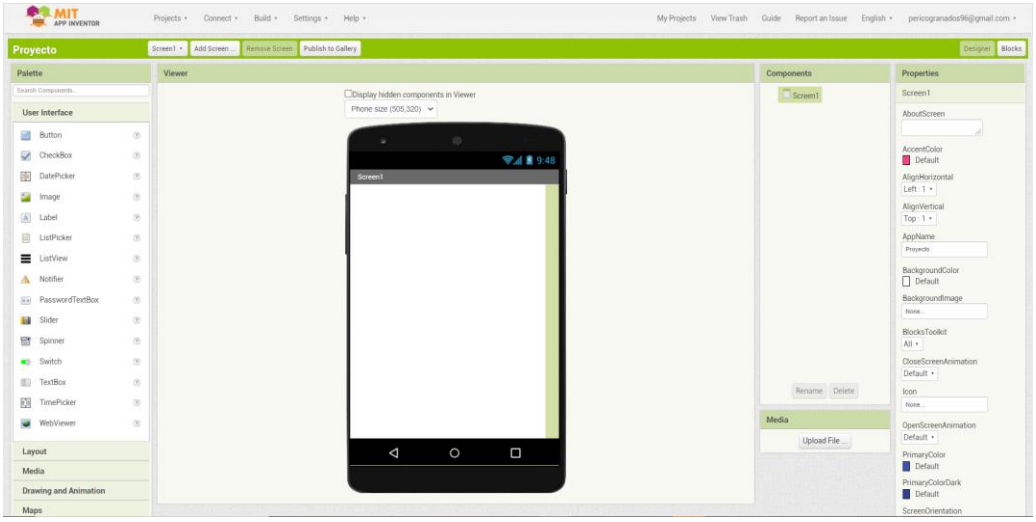


Figura 2-10: Vista de la interfaz del entorno *App Inventor*.

Una vez dada una introducción a ambos entornos de desarrollo, se van a presentar las diferencias entre ambos. El SO empleado en este caso ha sido Windows, debido a esto no se harán en ningún momento referencias a los SO de Linux o macOS.

Por un lado, Android *Studio* permite programar en su versión más actual en los lenguajes de programación admitidos por IntelliJ como son C++, Java o Kotlin (lenguaje que prefiere Google a la hora de desarrollar aplicaciones en Android). Por otro lado, *App Inventor* está programado en Java, pero a diferencia de Android *Studio* tiene la peculiaridad de que no se crea la aplicación a partir de lenguajes de programación de manera directa, sino que se hace a partir del enlace de bloques con funciones que se encargan de actuar de igual manera que lo hacen las funciones en los lenguajes de programación.

Otra de las diferencias entre ambas es que en el caso de Android *Studio* es un programa descargable que tiene que instalarse en el PC como cualquier otro programa más convencional mientras que en el caso de *App Inventor* es una aplicación web que no tiene descarga e instalación en PC, todo se hace directamente desde la web de cuyo mantenimiento se encarga el *MIT*.

En cuanto a la experiencia necesaria para el desarrollo de aplicaciones es necesario tener un conocimiento más experto para el caso de Android *Studio* ya que se necesitan conocimientos avanzados en los lenguajes de programación admitidos por el entorno, mientras que para *App Inventor* permite realizar aplicaciones para personas con conocimientos no tan avanzados en el mundo de la programación. Por estas razones, Android *Studio* permite realizar aplicaciones con una amplia variedad de posibilidades sin limitaciones en el desarrollo, mientas que en *App Inventor* las aplicaciones desarrolladas pueden llegar a estar un poco más limitadas, debido a la mayor simplicidad del entorno, pero las necesidades más utilizadas y básicas para los dispositivos móviles quedan cubiertas. Aunque puede parecer que puede haber una cierta desventaja de uso de *App*

Inventor con respecto a Android *Studio*, *App Inventor* ha generado desde su creación un gran número en cuanto a aplicaciones en Android se refiere porque aparte de ser más simple en su uso, sus aplicaciones se pueden añadir al centro de distribución principal de aplicaciones Android de Google, llamado Google *Play*, permitiendo a sus usuarios la libertad de compartir sus creaciones de manera libre y legal.

Para el diseño de aplicaciones para Android *Wear*, Android *Studio* permite el desarrollo de aplicaciones en este ámbito mientras que para el caso de *App Inventor* no es posible su desarrollo en este sentido. Por ello, para realizar una aplicación que debiese tener soporte adicional para el caso de un *smartwatch*, la única opción válida de las dos sería Android *Studio.*

En cuanto a las similitudes entre ambos se destacan que ambas son para desarrollos de aplicaciones exclusivas en Android. Ambas son herramientas *open-source* lo que hace que el número de usuarios que pueden acceder a ellas sea muy amplio. Ambas están reconocidas y aprobadas por Google para el desarrollo de aplicaciones en su plataforma de distribución oficial. Para el caso de la interfaz gráfica a la hora de diseñar las aplicaciones ambas permiten tener una gran variedad de funcionalidades y la forma de añadirlas es parecida para las dos opciones ya que con arrastrar los componentes que queramos podremos ir diseñando la interfaz lo que lo hace un trabajo más liviano e intuitivo.

Una vez vistos y estudiados los pros y los contras de cada IDE se decidió, que para la elaboración de la aplicación Android de este proyecto se iba a desarrollar en *App Inventor*. Por un lado, se decidió que debido a que los conocimientos para realizarla en Android *Studio* en un tiempo aceptable podrían no ser suficientes, unido a que las funcionalidades necesarias para realizarla estaban bien cubiertas en *App Inventor*, han sido las claves para elegir este entorno. Por lo tanto, a partir de ahora no se harán más referencias a Android *Studio* y todo aquello que tenga relación con la aplicación será en referencia a *App Inventor*.

2.2.4 Soldadura y otros elementos físicos

En este último punto se destacan aquellos elementos que han intervenido en la fase final del desarrollo del prototipo. Por un lado, la parte de la soldadura y por otro los elementos que han ayudado a la finalización de este prototipo.

En la parte de la soldadura, se han usado los elementos básicos que la llevan a cabo, en este caso un soldador convencional y estaño. No se ha necesitado estar en ningún entorno en especial como laboratorios o salas especiales, lo cual ha facilitado el trabajo en este proceso, ya que desde la habitación personal ha sido posible realizar este trabajo. El diámetro de los cables utilizados en esta etapa de soldadura es de 0.5 milímetros.

Se han visto diferentes tipos de tapas para la realización del prototipo con el suficiente tamaño para poder albergar sin problemas todos los componentes utilizados. Teniendo en cuenta que el tamaño de las *powerbanks* utilizadas varía en función de la capacidad de carga que tienen cada una, se han tenido en cuenta dos tipos de envases. Tenemos un pastillero semanal con tres tomas diarias para el caso de la *powerbank* más grande, mientras que para el caso de la *powerbank* más pequeña se han visto botes convencionales con una tapa de diámetro suficiente para poder albergar todos los elementos usados en el prototipo.

En este último punto debido a la dificultad de poder realizar este diseño, debido a la falta de medios y tiempo final para los últimos detalles, no se ha podido llevar a cabo en su totalidad.

El total de elementos necesitados por tapa se verá reflejado en la siguiente tabla 2-1:

	TAPA 1	TAPA 2
Controlador	Arduino UNO	Elegoo UNO R3
Módulo *bluetooth*	Un módulo HC-06	Un módulo HC-06
Pantalla	Un *display* OLED de 0.96”	Un *display* OLED de 0.96”
Conexión fte. de alimentación-Arduino	Un módulo TTL	Un módulo TTL
Fuente de alimentación	*Powerbank* de 10000mah	*Powerbank* de 5000mah

Tabla 2-1: Lista de componentes utilizados en cada tapa.

En la siguiente imagen (figura 2-11) se mostrarán algunos elementos que se han utilizado en esta última parte como el soldador o el estaño aplicado y un pastillero de referencia.

Figura 2-11: Vista los elementos finales utilizados.

3 Diseño y desarrollo de las tapas.

En este punto, se van a mostrar más en detalle los desarrollos y diseños llevados a cabo para los distintos elementos que componen este Trabajo de Fin de Grado. Para ello han sido fundamentales los entornos y herramientas descritas anteriormente en el punto 2.2.

En la elaboración de cada prototipo se han perseguido una serie de objetivos que afectan a distintos desarrollos ya sea en la parte de Arduino, de la aplicación Android o física. Como algún objetivo depende de la consecución de objetivos anteriores, se ha distribuido el trabajo en etapas.

Las etapas que se han seguido para la realización del proyecto han sido 3 que han establecido también el orden para poder llegar a alcanzar los objetivos principales propuestos. La primera etapa empieza por la elección de los componentes Arduino que una vez elegidos, se ha pasado a su programación en teniendo en cuenta el funcionamiento de estos y ayudándose en las conexiones del programa de diseño de componentes electrónicos Fritzing para una mejor visualización en esta memoria. Después de esta etapa se ha procedido a desarrollar la aplicación Android en *App Inventor* diseñando la interfaz pensando en un uso fácil e intuitivo, estableciendo una conexión entre la aplicación y los componentes por medio del módulo *bluetooth* perteneciente al prototipo con el que queramos conectar. Por último, se realizará la unión definitiva de todos los componentes físicos de manera que queden cohesionados en las tapas en las que se realice el diseño por medio de soldadura y creando un entorno cerrado para los componentes.

A continuación, se pasará etapa por etapa, viendo los diseños y desarrollos seguidos, que la unión de todos ellos resultará en el prototipo final que se tiene como objetivo final en la elaboración de este TFG.

3.1 Primera etapa: Diseño y desarrollo para los componentes Arduino

Este Trabajo de Fin de Grado tiene como primer objetivo la elaboración de un prototipo *hardware* en Arduino, cuyos componentes serán programados de manera que el *software* establezca las funcionalidades básicas para su correcto funcionamiento.

Esta primera se compone de todo el desarrollo seguido y el diseño correspondiente al apartado en Arduino. En este punto se irán mostrando los pasos que se han seguido a la hora de decidir qué dispositivos *hardware* van a intervenir en la elaboración de este TFG y para establecer las funcionalidades en el desarrollo correspondiente al *software* de los respectivos elementos elegidos.

Esta fase se compondrá de tres puntos en total, con un punto inicial en el que se explica el porqué de la decisión de cada elemento indicando la razón por la que ha sido elegido y la función que tiene en el prototipo. El segundo punto se centrará en el desarrollo de la programación de los dispositivos, en el que se mostrará el código desarollado indicando su funcionamiento y su diseño en los elementos que muestren información como en este caso sería la pantalla. El último punto tiene como objetivo mostrar las conexiones entre los dispositivos con la ayuda del programa anteriormente descrito Fritzing para establecer las

conexiones primero en un entorno digital, para mostrarlas de una manera más clara, antes de pasar a mostrar el entorno físico en el que se van a desarrollar las tapas.

3.1.1 Elección de los componentes

Como se ha empezado hablando al inicio de este punto, uno de los objetivos principales es el de proponer un dispositivo hardware que sea capaz de mostrar un contador progresivo mediante el cual se pueda tener un control sobre el tiempo necesario de espera para realizar una toma de un medicamento o cualquier otro suplemento que necesite de un control horario.

Una vez establecido el objetivo principal a alcanzar, se han estudiado qué posibilidades hay dentro de lo que se ofrece en el mercado en lo correspondiente a los dispositivos digitales de Arduino con capacidades para llegar a ofrecer aquellas funcionalidades que se pretenden conseguir en este proyecto. No obstante, y como es lógico, también se han tenido en cuenta mis conocimientos personales que se han ido adquiriendo tanto en el grado, como en la realización de proyectos personales.

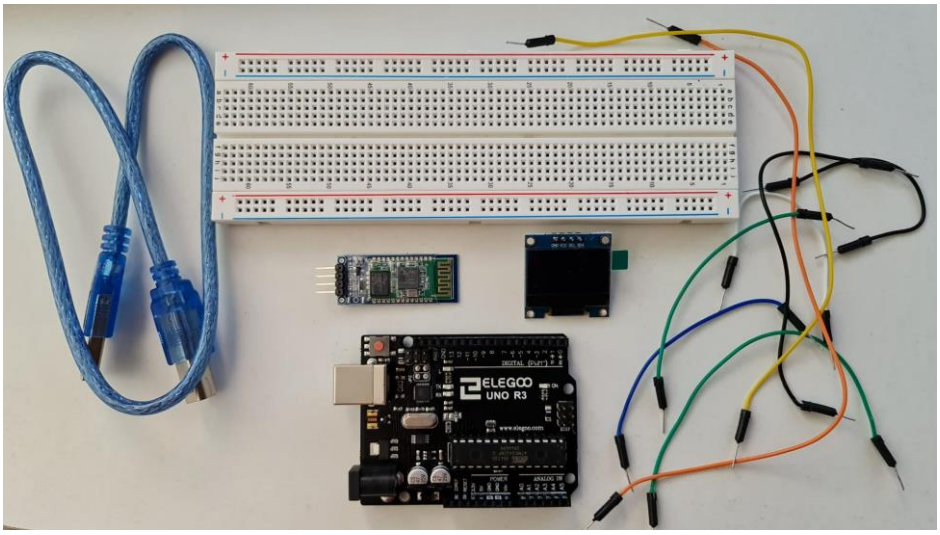


Figura 3-1: Vista de los componentes que intervienen en la primera etapa de Arduino.

Teniendo en cuenta lo que se necesita y las capacidades, el elemento principal en el desarrollo del prototipo es la placa de desarrollo hardware que tiene como elementos más importantes 14 pines digitales de *I/O*, 6 entradas analógicas, conexión USB, un botón de *reset* y un conector de alimentación. A partir de aquí saldrán las conexiones al resto de elementos y se pasará el código con la funcionalidad de cada uno.

Se establecerá un enlace entre la tapa y el *smartphone,* que se encargará de transmitir la información para conseguir la sincronización de tiempo con la pantalla de la tapa. Para estas acciones, el módulo *bluetooth* que cumplirá esta función es el denominado HC-06 programable con Arduino que se compone de los pines de alimentación VCC que funciona de 3.3 a 5V, el pin de tierra GND y por último los dos pines que se encargan de transmitir la información llamado *TX* y el receptor *RX*.

El componente encargado de representar la información del contador y la fecha por pantalla es un *display* OLED de 0.96 pulgadas. Admite como el módulo HC-06 un voltaje de alimentación de 5V y sus ventajas frente a las pantallas LCD es que no necesitan de una

iluminación *backlight* ya que solo se encienden los píxeles necesarios que han sido programados por código.

El conjunto de estos componentes define la parte física de las tapas que son capaces de conseguir alcanzar objetivo final, a falta de establecer una fuente de alimentación válida para los componentes y con la suficiente autonomía para ofrecer un tiempo de vida suficientemente aceptable.

3.1.2 Programación de los componentes

Este segundo punto tratará en detalle el código que se ha desarrollado para establecer las funcionalidades básicas de cada componente en función de la conexión a los pines de la placa de desarrollo Arduino Uno.

A continuación, se mostrará el código entero que se ha compilado y subido a la placa que ha permitido el correcto funcionamiento en el *hardware* de este proyecto. Se explicará que funcionalidades se establecen, haciendo alusión al conjunto de líneas que sean necesarias en cada caso y que son determinantes para el objetivo final:

```
1.  #include <U8glib.h>
2.  #include <stdlib.h>
3.  #include "SoftwareSerial.h"
4.  #include "string.h"
5.
6.  SoftwareSerial bluetooth(2, 3); //(RECEPTOR, TRANSMISOR)
7.
8.  int Horas = 0;
9.  int Minutos = 0;
10. int Segundos = 0;
11. int Dia = 0;
12. int Fecha = 0;
13. int Mes = 0;
14. int Anno = 0;
15.
16. char FechaBuffer[30];
17. char* VariableTiempo[6];
18. char* VariableSeparado;
19.
20. String Meses;
21. String Bl_Activo;
22.
23. static unsigned long seg = 0;
24.
25. U8GLIB_SSD1306_128X64 u8g(U8G_I2C_OPT_NO_ACK | U8G_I2C_OPT_FAST);
26.
27. void setup(void) {
28.   Serial.begin(9600);
29.   bluetooth.begin(9600);
30.
31.   delay(1000);
32. }
33.
34. void loop() {
```

```
35.   if (bluetooth.available() > 0)
36.   {
37.     Bl_Activo = bluetooth.readString();
38.
39.     if (Bl_Activo.startsWith("1"))
40.     {
41.       Bl_Activo.toCharArray(FechaBuffer, Bl_Activo.length() + 1);
42.
43.       VariableSeparado = strtok(FechaBuffer, ",");
44.       for (int i = 0; i <= 6; i++) {
45.         VariableTiempo[i] = strtok(NULL, ",");
46.         Serial.print(VariableTiempo[i]);
47.         delay(10);
48.       }
49.
50.       Dia = atoi(VariableTiempo[0] - 1);
51.       Fecha = atoi(VariableTiempo[1]);
52.       Mes = atoi(VariableTiempo[2]);
53.       Anno = atoi(VariableTiempo[3]);
54.       Serial.print(Dia - 1);
55.       Serial.print(Fecha);
56.       Serial.print(Mes);
57.       Serial.print(Anno);
58.
59.       Horas = atoi(VariableTiempo[4]);
60.       Minutos = atoi(VariableTiempo[5]);
61.       Segundos = atoi(VariableTiempo[6]);
62.       Serial.print(Horas);
63.       Serial.print(Minutos);
64.       Serial.print(Segundos);
65.     }
66.   }
67.   Contador_Y_Fecha();
68.   delay(10);
69. }
70.
71. void Contador_Y_Fecha(){
72.
73.   if (millis() - seg > 950) {
74.       seg = millis();
75.       Segundos++;
76.   }
77.
78.   if (Segundos > 59) {
79.       Segundos = 0;
80.       Minutos++;
81.   }
82.
83.   if (Minutos > 59) {
84.       Minutos = 0;
85.       Horas++;
86.   }
87.
88.   if (Horas > 23) {
```

```
89.       Horas = 0;
90.       Minutos = 0;
91.       Fecha += 1;
92.   }
93.
94.     switch (Mes) {
95.       case 01:
96.         Meses = "Enero";
97.         break;
98.       case 02:
99.         Meses = "Febrero";
100.              break;
101.           case 03:
102.             Meses = "Marzo";
103.             break;
104.           case 04:
105.             Meses = "Abril";
106.             break;
107.           case 05:
108.             Meses = "Mayo";
109.             break;
110.           case 06:
111.             Meses = "Junio";
112.             break;
113.           case 07:
114.             Meses = "Julio";
115.             break;
116.           case 8:
117.             Meses = "Agosto";
118.             break;
119.           case 9:
120.             Meses = "Septiembre";
121.             break;
122.           case 10:
123.             Meses = "Octubre";
124.             break;
125.           case 11:
126.             Meses = "Noviembre";
127.             break;
128.           case 12:
129.             Meses = "Diciembre";
130.             break;
131.         }
132.       Pintar_pantalla();
133.     }
134.
135.     void Pintar_pantalla() {
136.
137.       u8g.firstPage();
138.       do
139.       {
140.         u8g.setFont(u8g_font_fur17r);
141.
142.         String strCont = String("");
```

```
143.          char time[10];
144.          if (Horas < 10)
145.            strCont += "0";
146.          strCont += Horas;
147.          strCont += ":";
148.          if (Minutos < 10)
149.            strCont += "0";
150.          strCont += Minutos;
151.          strCont += ":";
152.          if (Segundos < 10)
153.            strCont += "0";
154.          strCont += Segundos;
155.          strCont.toCharArray(time, 10);
156.
157.          time[12] = 0x00;
158.          u8g.drawStr(20, 40, time);
159.
160.          u8g.setFont(u8g_font_7x14r);
161.          u8g.setPrintPos(25, 55);
162.          u8g.print(Fecha);
163.          u8g.setPrintPos(40, 55);
164.          u8g.print("/");
165.          u8g.setPrintPos(50, 55);
166.          u8g.print(Mes);
167.          u8g.setPrintPos(70, 55);
168.          u8g.print("/");
169.          u8g.setPrintPos(80, 55);
170.          u8g.print(Anno);
171.
172.        }
173.        while (u8g.nextPage());
174.      }
```

En primer lugar, de la línea 1 a la 4, establecemos las librerías que son esenciales ya que contienen las funciones y elementos de diseño necesarios para la correcta programación de cada dispositivo. La librería U8glib.h es especial de *displays* monocromáticos ya sean TFTs u OLEDs que en nuestro caso ha servido para establecer el de diseño a la hora de mostrar por pantalla la información necesaria. Con la librería *Software Serial* se ha conseguido permitir y establecer la comunicación con los pines digitales de la placa de desarrollo de Arduino. Con la librería *String* establecemos cadenas de alfanuméricos para poder tener cadenas de objetos, que permitan programar conjuntos como la fecha o el tiempo transcurrido. Por último, la librería *Stdlib* establece las funciones correspondientes de la programación C tales como macros o los distintos bucles utilizados.

En la línea 6 de código establecemos la conexión con la transmisión y recepción de información del módulo *bluetooth*, con los pines digitales 2 (receptor) y 3 (transmisor) de la placa de desarrollo que se encargarán de realizar el trabajo de recibir aquella información que transmitamos del *smartphone* y viceversa.

De la línea 8 a la 14 establecemos las variables correspondientes al tiempo para establecer el contador, inicializando las variables de horas, minutos y segundos en las líneas que van de la 8 a la 10 y en el caso de la fecha sus variables se inicializan en las líneas desde la 11 a la 14 con el día, fecha, mes y año actuales. De la línea 16 a la 18 se inicializan las variables

que van a permitir alternar y separar las variables del tiempo más adelante como por ejemplo en el caso de la fecha para establecer los días, meses y años. En la línea 20 inicializamos una cadena que utilizaremos para establecer los meses en un bucle más adelante y en la 21 tenemos una cadena que utilizaremos para establecer las comunicaciones con el bluetooth y podamos pasar la información que deseemos. Por último, la variable estática de la línea 23 es la que usaremos para el bucle con el que iniciaremos y contaremos los segundos.

En la línea 25 establecemos la función característica del diseño en particular para los *bytes* que tenemos por pantalla que como se especificó en el punto 2 debido a que su resolución es de 128*64.

En las líneas que van de la 27 a la 32, inicializamos la conexión serie y el módulo *bluetooth*.

Desde las líneas 34 a la 69, se establece el bucle principal en el que de manera resumida estableceremos que una vez conectado con el módulo *bluetooth* separaremos la cadena para tener cada variable en la que luego escribiremos según sus funciones.

Para las líneas que van de la 35 a la 39 se establecen los bucles que permiten que una vez se ha detectado y conectado el *bluetooth*, podamos empezar con el objetivo establecido en cuanto a las funcionalidades.

En lo respectivo a las líneas que van desde la 40 a la 48, estableceremos el bucle que nos permitirá ir pasando desde las variables correspondientes a la fecha, a las variables del contador, realizando una separación previa de estas variables con la función strtok. Las variables que tienen la fecha pasan de ser variables de *arrays* de caracteres a tipo numérico entero (int) por medio de la función atoi y posteriormente se imprimen en el puerto serie como aparece representado desde las líneas 50 a la 57. Lo mismo pasa para el caso del contador de tiempo que en este caso aparece desde las líneas 59 a la 64.

Para finalizar la función del primer bucle, en la línea 67 llamaremos a la siguiente función llamada “Contador_Y_Fecha” la cual se encarga de establecer que funciones se deben realizar cuando el tiempo está contando y de alternar entre los distintos meses.

La función a la que se acaba de hacer referencia se desarrolla desde las líneas 71 a la 133. En la primera parte establecemos el aumento de los segundos desde las líneas 73 a la 76 en la que hacemos uso de la variable estática anteriormente inicializada “seg” y de los milisegundos (*millis*) que es la variable que permite tener en cuenta el tiempo en Arduino. Desde las líneas 78 a la 81 cuando el tiempo en segundos llega a 60 sumaremos el contador de minutos el valor de +1 y volvemos a poner el contador de segundos a 0. Lo mismo pasa para el caso en que el contador de minutos llega a 60 en el cual aumentaremos en este caso el contador de las horas en +1 y los minutos y segundos volverán al valor inicial de 0, esto aparece programado desde las líneas 83 a la 86. Para la finalización de la funcionalidad básica de esta función, desde las líneas de código 88 a la 92, se ha programado que cuando el número de horas llega a 24 el contador de fecha se ve aumentado en +1, mientras que el contador volverá en todos sus valores a 0.

Para establecer los meses se ha programado un bucle *while* que irá recorriendo los distintos meses desde enero hasta diciembre en un total de 12 casos, desde las líneas 94 a la 131.

En la línea 132 se establece la llamada a la última función del apartado de *software* de Arduino cuya funcionalidad pasará a explicarse a continuación que pintará los resultados por pantalla.

Entre las líneas 135 y 174, se ha programado la última función la cual se encarga de representar en la pantalla el contador y la fecha teniendo en cuenta las funcionalidades anteriormente programadas. A continuación, se va a explicar que funciones establecen el diseño en la pantalla OLED.

En esta función establecemos primero una página que será la única que mostrará por pantalla los objetivos previstos en este apartado, que definiremos en su inicio en la línea 137 y finalizaremos en la línea 173.

Es a partir de la línea 138 donde empieza la programación encaminada al diseño de lo que se va a imprimir por pantalla. En primer lugar, se ha pasado a definir el contador progresivo, que empieza en la línea 140 donde establecemos la fuente y su tamaño que es requisito previo antes de pintar los caracteres. Primero definimos la cadena donde vamos a escribir todos los elementos que corresponden al contador en la línea 142 y los valores numéricos se definirán en la 143 con la variable "*time*". Desde las líneas 144 a la 158 se establece diseño del contador que tiene el formato "HH:MM:SS" empezando en la línea 144 con las horas y finalizando en la 147, en segundo lugar están los minutos desde la línea 148 a la 151 y por último los segundos desde la línea 152 a la 154. Se almacenan todos los caracteres numéricos de las variables en un *array* en la línea 155. Por último, para representar por pantalla establecemos el formato y el punto de inicio de escritura en las líneas 157 y 158.

Posteriormente se imprime la fecha desde las líneas 160 a la 170 en el formato de "DD/MM/AAAA" estableciendo en primer lugar como en el caso del contador la fuente, que será de tamaño un poco más pequeña que el contador. En segundo lugar establecemos en la primera posición el día en las líneas 161 y 162 estableciendo después la primera barra separadora en las líneas 163 y 164. Posteriormente se realiza el mismo procedimiento en las líneas 165 y 166 con el mes y en las líneas 167 y 168 la segunda barra. Por último, para finalizar la parte correspondiente a representar la fecha es la que corresponde al año, en las líneas 169 y 170.

Con esto finaliza la parte correspondiente al desarrollo software correspondiente a todos los elementos que intervienen en este proyecto. Una vez terminado el desarrollo *software*, se ha pasado al último apartado en lo correspondiente al diseño y desarrollo de los componentes físicos de Arduino.

3.1.3 Diseño y conexión física de los componentes

Una vez programado el código expuesto en el punto anterior se ha pasado a trabajar con los elementos físicos para comprobar que todo funciona de manera correcta sin entrar en muchos más detalles, ya que en este punto solo queremos de momento poder ir comprobando su funcionamiento junto con la aplicación que se ha ido desarrollando y que explicaremos en detalle en el punto 3.2.

En este último apartado del desarrollo y diseño de Arduino, se han utilizado por un lado la herramienta de diseño en PC para ofrecer una mejor visibilidad de las conexiones entre los

componentes y por otro lado utilizaremos los componentes físicos para realizar estas mismas conexiones en un entorno real.

Para este paso, primero se ha usado la herramienta de diseño con componentes Arduino Fritzing que ha servido de ayuda para mostrar el diseño de la unión de los componentes (figura 3-2).

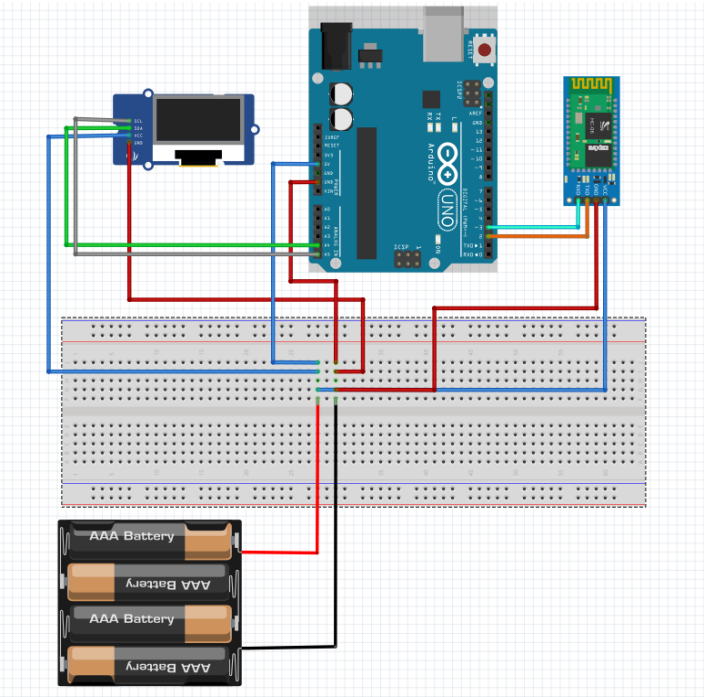


Figura 3-2: Diseño en Fritzing.

Ahora es el turno de la unión de los distintos componentes con los pines que corresponden para cada uno de ellos y como se ha visto diseñado en la herramienta de PC. Se mostrará una visión global del primer montaje en la figura 3-3.

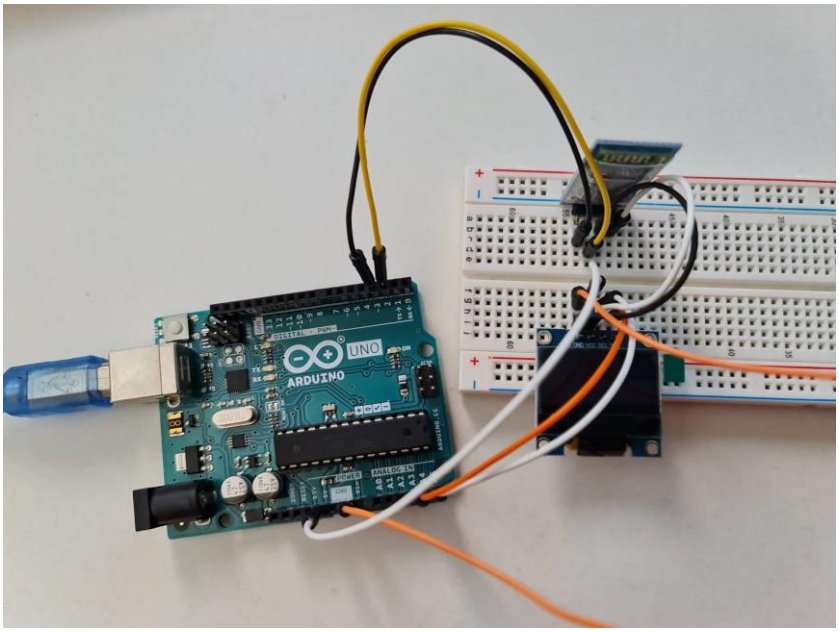


Figura 3-3: Vista del entrono físico de la conexión de los componentes a la *protoboard*.

Con esto habría terminado la parte que corresponde a Arduino, tanto *software* como *hardware*, y ya una vez comprobado que el módulo *bluetooth* ha conectado correctamente con el *smartphone* pasamos a la siguiente etapa de este punto que trata el desarrollo y diseño de la aplicación Android.

3.2 Segunda etapa: Diseño y desarrollo de la aplicación Android

Esta segunda etapa pretende mostrar en detalle el desarrollo que se ha seguido para la creación de la aplicación Android con las funcionalidades necesarias para conseguir los objetivos que se establecen para este punto del TFG, conectando un dispositivo móvil a una o varias tapas.

Como se ha explicado anteriormente, en el punto 2.2 el entorno utilizado para el desarrollo de la aplicación es *App Inventor* que nos ha permitido diseñar y ordenar elementos añadidos como botones o textos en la interfaz y establecer todas las funcionalidades necesarias para establecer una buena conexión con las tapas por medio de los módulos *bluetooth*, y poder sincronizar la información de nuestro *smartphone* a la pantalla.

Para mostrar el desarrollo de la aplicación iremos mostrando poco a poco cada bloque creado, que programa funcionalidades tales como conexiones, alarmas o sincronizaciones. En lo relativo al diseño se explicará a continuación el orden y establecimiento de los distintos elementos dispuestos mostrando como ejemplo una pantalla genérica de un dispositivo Android (figura 3-4).

En primer lugar, tendremos los botones de selección iniciales en los que conectaremos con las tapas que tengamos a nuestra disposición. En este caso, en primera instancia, serán un total de dos tapas que se distinguirán por los colores azul y naranja para ayudar a ver su distinción entre ambas, aunque se puedan sincronizar a la vez con nuestro *smartphone*.

En segundo lugar, con un mismo botón de selección rojo, podremos desconectar las tapas a las que estemos conectadas en caso de que no necesitemos el uso de alguna de ellas.

El botón que aparece en verde sirve para reiniciar el tiempo del contador de todas las pantallas que tengamos en el momento del *click* conectadas a nuestro *smartphone*. En caso de que el usuario realice la misma toma desde distintas tapas, esto le será de utilidad para tener en cuenta el mismo tiempo transcurrido en todas las tapas.

Una funcionalidad parecida a la que se acaba de explicar se establece en los siguientes botones representados en los mismos colores que los que se representan en las conexiones con las tapas, pintando en azul la primera tapa conectada y en naranja la segunda tapa conectada. La funcionalidad es parecida, ya que se trata de una sincronización de tiempo, pero en este caso individualmente para cada tapa por si el usuario realiza una toma distinta para cada tapa.

En los apartados de “HORA ACTUAL” y “DÍA DE LA SEMANA” se representa información que se considera interesante como son la hora actual y el día en el que nos encontramos en caso de que el usuario necesite saber esta información para tomas semanales.

Los botones que se representan como “Toma 1” a la “Toma 8” son botones de selección horaria que se utilizan para programar hasta un total de 8 alarmas ya que dependerá de

cada usuario o incluso puede existir el caso de que se compartan varias tomas para varios usuarios en los mismos botes donde puedan tener sus medicaciones.

A continuación, en la figura 3-4 se mostrará la interfaz de la aplicación con todas las funcionalidades propuestas en este trabajo que permite alcanzar los objetivos que hemos marcado para esta segunda etapa:

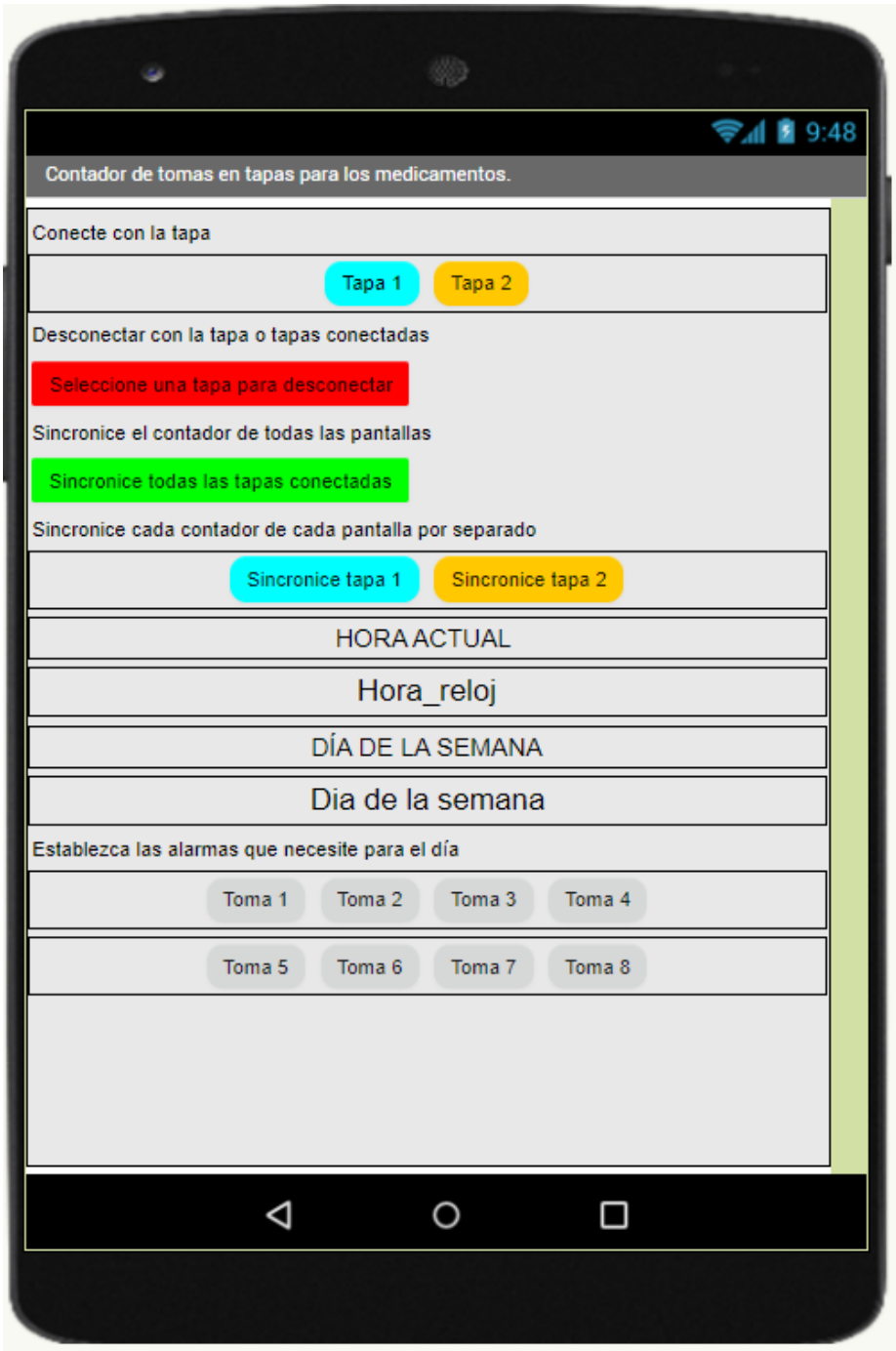


Figura 3-4: Vista de los distintos elementos de la aplicación en la interfaz de la aplicación web.

Como se puede apreciar en la figura 3-4 el diseño de la interfaz de aplicación visto en un *smartphone* Android sería así, teniendo en cuenta que imita a un *smartphone* genérico y

que siempre dependerá del tamaño de cada uno. En la siguiente figura se puede apreciar la lista de elementos no visibles.

Los elementos no visibles que aparecen en la figura 3-5 forman una parte de lo que a continuación se va a explicar sobre la funcionalidad de cada botón. Son funciones a las que llamaremos para poder combinar los elementos de la interfaz con la información y funcionalidades internas de nuestro *smartphone*.

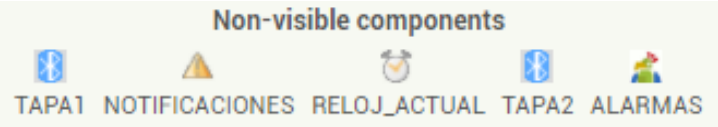


Figura 3-5: Lista de componentes no visibles utilizados.

En el caso de los componentes representados con el símbolo de *bluetooth*, llamados "TAPA1" y "TAPA2", se utilizarán para interaccionar con los módulos bluetooth de Arduino a los que enviaremos la información de nuestro *smartphone* correspondientes al tiempo que representaremos en las tapas.

En el componente llamado "NOTIFICACIONES" se almacenan las funciones y llamadas a los tipos de notificaciones internas que tiene cada *smartphone* en sus funcionalidades internas. Los elementos como las alertas que aparecerán representadas con un texto con sus correspondientes elementos de diseño formarán parte de este proyecto.

Para el caso del elemento llamado "RELOJ_ACTUAL" se cogerá el tiempo real del reloj interno del *smartphone* que estemos utilizando para poder realizar una sincronización correcta con las tapas. También se usará para programar las alarmas necesarias comparando el tiempo real con el tiempo establecido por la alarma.

El último elemento no visible es una extensión externa gratuita (*descargada del repositorio de* https://puravidaapps.com/alarm.php) la cual permite relacionar el reloj interno del *smartphone* junto con las funcionalidades básicas de *App Inventor*. Establecer una alarma con este componente permite coger también las funcionalidades propias de las alarmas de los dispositivos móviles como son las de vibración y sonido, pudiendo establecer el tono de alarma que desee el usuario. En primera instancia, se había programado una alarma con los elementos iniciales internos de la aplicación web con otros componentes no visibles como son las funcionalidades de sonido, vibración y hasta 8 relojes adicionales para cada alarma. Pero con relación a la solución final se ha podido obtener una mejor optimización de los elementos de la app ya que no se necesitan tantos componentes y permite una cómoda ejecución de la aplicación en segundo plano aumentando su facilidad de uso.

Una vez visto el diseño y los componentes no visibles que hacen uso de las funcionalidades internas de los *smartphones*, pasaremos a ir explicando bloque por bloque el funcionamiento de cada botón y cada funcionalidad vista en la interfaz. En la Figura 3-6 aparece la inicialización del reloj actual en la aplicación.

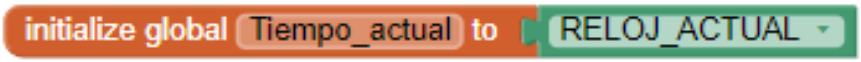


Figura 3-6: Bloque de inicialización del reloj de la aplicación.

En los dos bloques representados en la figura 3-7 se puede apreciar el funcionamiento de los primeros botones representados con fondo azul y naranja en la interfaz, cuyas funciones son la de conectar con las tapas que estén disponibles. La primera parte, correspondiente al bloque "*BeforePicking*" establece los elementos disponibles para la conexión en una lista, donde seleccionaremos la tapa a la queramos conectarnos. En el bloque "*AfterPicking*" se establece la funcionalidad una vez seleccionado el dispositivo, que se encarga de poner un mensaje en el *smartphone* indicando que se ha conectado correctamente a la tapa, estableciendo el fondo de color negro y el texto en azul. Para los botones similares de conexión con más tapas se establecerán los bloques con las mismas funcionalidades llamando en cada caso al botón correspondiente.

Figura 3-7: Bloques de funcionalidad de los botones de conexión con cada tapa.

De manera parecida a los bloques anteriormente descritos, el botón de desconexión de tiene una funcionalidad similar (funcionalidad en la figura 3-8). En este caso el funcionamiento para desconectar de cada tapa se realiza desde el mismo botón, en el que seleccionaremos cada vez en una lista de dispositivos bluetooth aquellos de los que queramos desconectarnos. La primera parte de la funcionalidad que corresponde al apartado "*BeforePicking*" al igual que en los botones anteriores establece la lista de los componentes disponibles bluetooth.

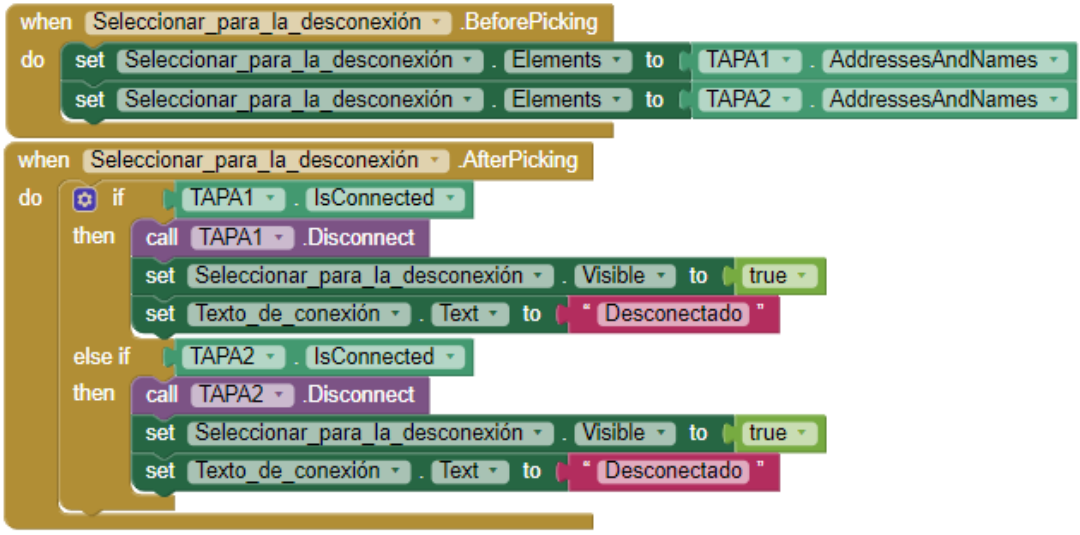


Figura 3-8: Bloques de funcionalidad del botón de desconexión de las tapas.

La funcionalidad cambia con respecto a lo anterior en el apartado "*AfterPicking*" en el cual una vez pulsado el dispositivo del que queramos desconectarnos, chequea por medio de un bucle *if* si ese dispositivo está conectado a nuestro *smartphone* y si es así llamará al módulo *bluetooth* de la tapa para que se desconecte dejándolo visible y saltando un mensaje de confirmación de que se ha desconectado correctamente de la tapa, el cual aparecerá reflejado en la pantalla del *smartphone*.

En la figura 3-9 se encuentra la funcionalidad que se corresponde con el botón representado en verde en la interfaz que tiene el propósito principal de establecer el tiempo del contador a ceros, para que una vez realizada una toma si esta es común en todas las tapas conectadas, se pueda sincronizar el tiempo de toma en todas las tapas. También establece como propósito secundario, establecer la fecha actual que se enviará a la pantalla de la tapa gracias al envío de esta información al módulo *bluetooth*.

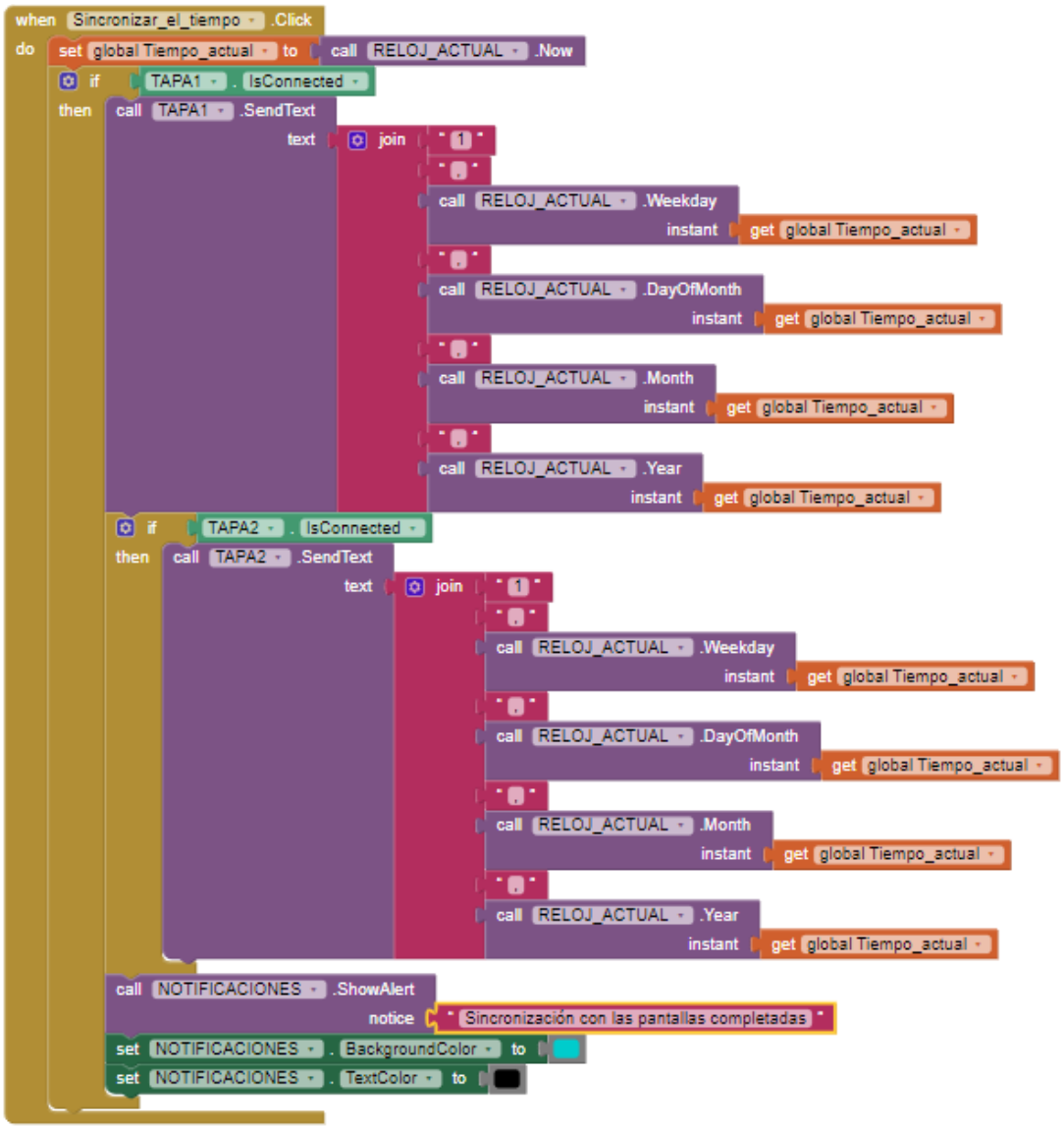


Figura 3-9: Bloque de funcionalidad del botón de sincronización de todas las tapas.

La funcionalidad del botón consiste en que, al hacer *click* primero coge como referencia el tiempo actual que contiene tanto la fecha como el tiempo de reloj que serán los principales elementos que tienen participación en este botón. Una vez cogido el tiempo del reloj,

comprueba con un bucle *if* que la tapa 1 a la que nos hayamos conectados esté activa y si se cumple esta condición, enviará la información del reloj correspondiente al día actual con las llamadas "*Weekday*" y "*DayOfMonth*", también hará lo mismo para el mes actual con la llamada "*Month*" y por último el año será enviado por la llamada "*Year*". En el caso de la hora no especificaremos nada ya que como lo que queremos que el tiempo se reinicie, al no especificar el envío de la hora, esta reseteará el tiempo del contador de forma automática. Todas estas llamadas cogerán el tiempo con la variable correspondiente al tiempo actual con la cual hemos inicializado el reloj. Al terminar con la primera tapa, realizará el mismo chequeo con la segunda tapa y si se cumple que está conectada realizará el mismo procedimiento que para la anterior. Por último, una vez finalizado este proceso, saltará una notificación de confirmación de sincronización con las tapas.

El bloque correspondiente a la figura 3-10 tiene la funcionalidad de los botones de sincronización del tiempo en las tapas, pero en este caso se realiza en cada tapa conectada por separado. En el ejemplo, el botón corresponde a la sincronización de la primera tapa a la que nos hayamos conectado.

El funcionamiento de estos botones es muy similar al botón que se ha descrito de la figura 3-9 pero en este caso se realiza la sincronización de cada tapa con dos botones que se corresponderán con los mismos colores de conexión iniciales de cada una. Para ello volvemos a coger el tiempo actual, accediendo al tiempo del reloj interno del *smartphone*. Una vez tenemos el reloj chequeamos que la tapa que queremos actualizar esté conectada y en caso de que así sea, escribiremos la fecha de igual manera que para la sincronización del botón anterior y también sin pasar la hora ya que como queremos establecer el contador desde cero se reiniciará automáticamente como en el anterior caso. Por último, el mensaje de confirmación de que se ha sincronizado correctamente con la pantalla de la tapa será en específico de la tapa que necesitemos sincronizar.

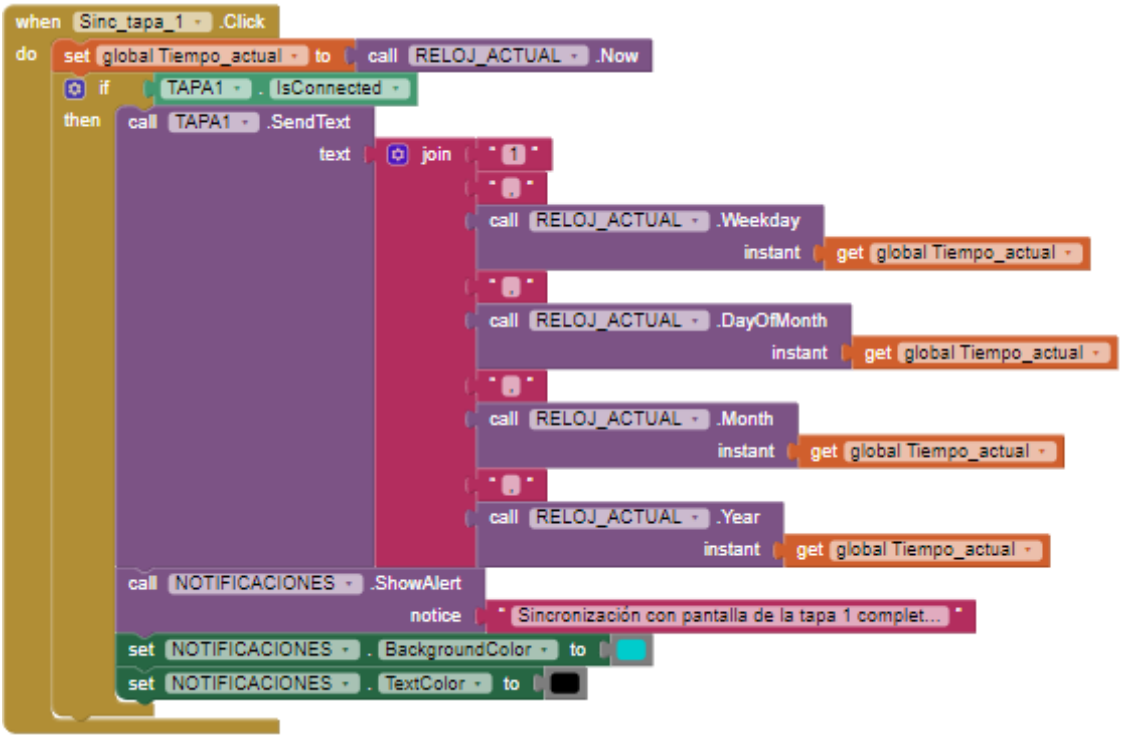


Figura 3-10: Bloque de funcionalidad del botón de sincronización de una tapa.

Como se puede apreciar en el diseño de la interfaz de la aplicación se pretende facilitar más información que sirva de ayuda para la toma, que corresponden a la parte de la "HORA ACTUAL" y "DÍA DE LA SEMANA". La primera parte pretende servir como complemento al contador progresivo de la tapa para tener una relación con del tiempo actual. La segunda parte establece el nombre del día actual, para ayudar a que si son tomas semanales no se tenga la necesidad de recordar el día en específico en el que se tenga que realizar dicha toma (figura 3-11).

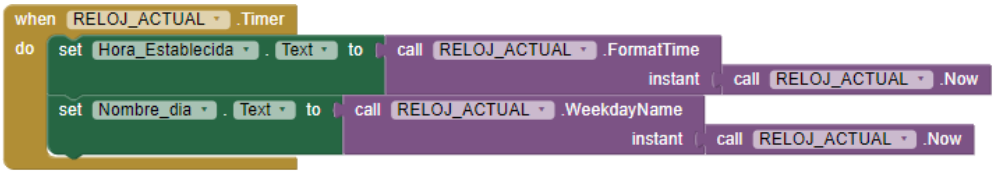


Figura 3-11: Bloque de funcionalidad que representa la hora y el día de la semana actuales.

Para programar la alarma de las tomas, se han creado un total de 8 bloques con la misma funcionalidad. En la figura 3-12 aparecerá un bloque de ejemplo utilizado para la primera alarma y posteriormente se explicará su funcionalidad.

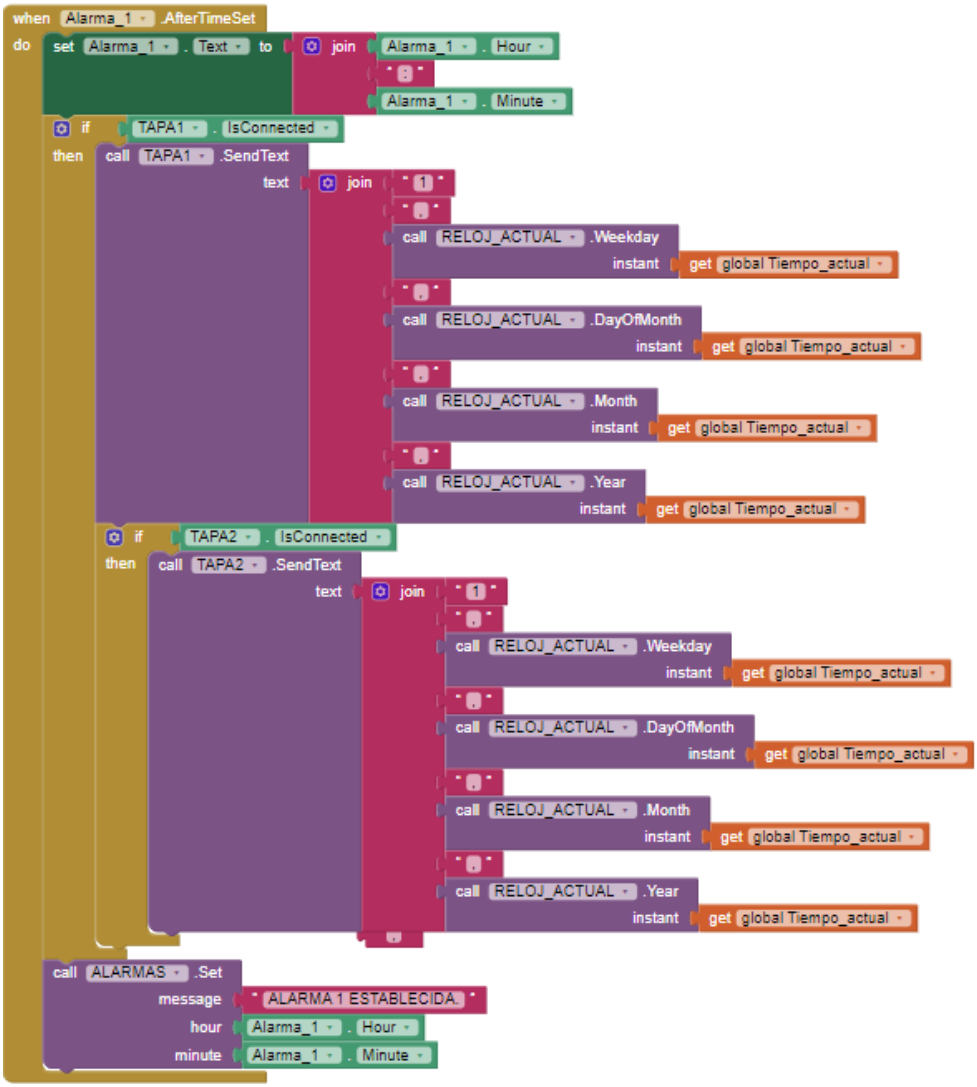


Figura 3-12: Bloque de funcionalidad del botón de programación de la alarma.

Para establecer una alarma, primero se hace *click* en el botón, en el cual se abrirá un cuadro de selección horaria que selecciona el momento en el que se debe realizar la siguiente toma y una vez elegido pinta la hora en el mismo botón. Una vez seleccionado el momento de la toma se realizará una comprobación de conexión mediante un bucle *if* en el que se detectará si la primera tapa está conectada a nuestro dispositivo móvil. En caso de que la tapa esté correctamente conectada, actualizará la fecha actual en la pantalla al igual que reiniciará el tiempo del contador. Posteriormente se realizará el mismo procedimiento para la segunda tapa en caso de estar esta conectada. Por último, se establece la llamada la extensión descargada, haciendo uso de una de sus funciones la cual establece una función en la que la hora que hemos seleccionado se quedará guardada en el reloj interno en la sección de alarmas. Estas alarmas se quedarán guardadas con un título indicando que alarma se ha creado y la hora que se ha decidido para la toma.

El bloque programado que aparece representado en la figura 3-13 guarda relación con el anteriormente descrito de la figura 3-12 ya que forma parte de la funcionalidad de los botones de selección de horaria para la programación de la alarma. Aunque aparezca un bloque de ejemplo para la primera alarma (como en el caso anterior), el funcionamiento de ambos está regido por cada uno de los 8 botones de selección de alarma y como sucede para el caso anterior, cada botón tiene su bloque programado. El funcionamiento es simple, su función es la de que cuando se vuelva a pulsar el botón y no se seleccione una nueva hora queriendo cancelar su uso, el botón pasará a tener su nombre inicial.

Figura 3-13: Bloque de funcionalidad del botón de programación de la alarma al hacer *click* una vez establecida la alarma.

Una vez terminada la aplicación y comprobado que ha ido bien con la funcionalidad desarrollada y diseñada de Arduino, se ha dado por finalizada la elaboración de la funcionalidad completa del prototipo y se ha pasado a la tercera etapa y final.

3.3 Tercera etapa: Soldadura y elementos finales

Una vez programados los componentes de Arduino y desarrollada la aplicación, la etapa final consistirá en añadir los elementos definitivos para la finalización del prototipo. Para ello se han necesitado dos fuentes (para las dos tapas) de alimentación en forma de baterías portátiles, para que puedan ser dispositivos independientes y autónomos que no requieran de estar alimentados en un punto fijo, como lo hacían hasta ahora por sus conexiones USB al PC. Estas permiten una recarga sencilla y estar informados en todo momento de cuanta batería tenemos por medio de las luces led de las *powerbanks* utilizadas. Para conseguir establecer la alimentación en los prototipos, hemos utilizado dos módulos TTL usando sus pines de 5V y tierra. Una vez tengamos estos elementos, ya haremos uso de la soldadura y encaje de los pines para que, una vez terminado este proceso, podamos agregar el dispositivo completo a las tapas que sean capaces de albergarlo en su totalidad.

En esta etapa final del desarrollo del prototipo se ha aplicado la soldadura como el elemento de unión final de los componentes *hardware* que han intervenido en el proyecto y que ayuda a la unificación de todos ellos de manera más fiable. Para ello se han hecho uso

de varios cables de 0.5 mm de diámetro, que han sido los elementos de unión, los cuales por medio del uso de un soldador convencional aplicando estaño a los pines de los componentes, hemos podido unificar de manera adecuada el *hardware* completo, a excepción de las placas de desarrollo en los que hemos usado directamente las entradas de sus pines para esta conexión por medio del encaje (Ejemplo real en la figura 3-14).

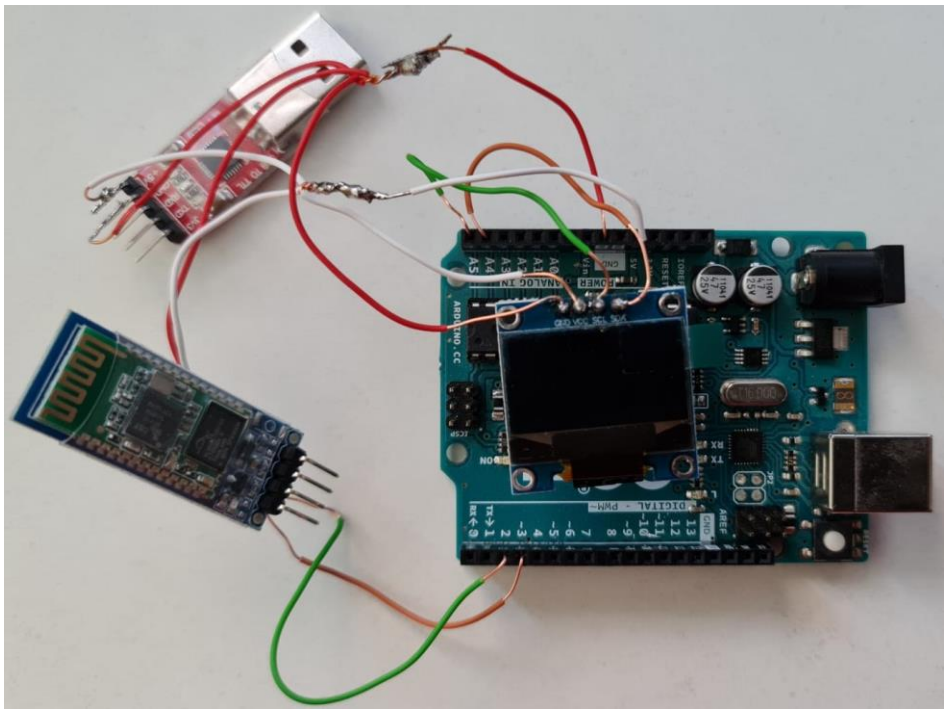


Figura 3-14: Vista los elementos finales unificados.

4 Pruebas realizadas en la elaboración del dispositivo

Como se ha ido explicando en el punto anterior, este Trabajo de Fin de Grado se ha ido realizando en un total de 3 etapas de diseño y desarrollo del prototipo, en las cuales se han ido haciendo algunas pruebas que han servido también como requisito previo a pasar a la siguiente etapa, aparte de comprobar que el funcionamiento es correcto. Dentro de cada etapa se han ido realizando pruebas también para intentar evitar arrastrar posibles errores que luego pudiesen ser más difíciles de localizar.

4.1 Pruebas realizadas en la etapa de Arduino

Como se ha visto en el punto 3, la primera etapa se ha basado en el *hardware* y *software* para establecer la funcionalidad con los objetivos propuestos en cada uno de los elementos que han intervenido para el desarrollo del prototipo en Arduino.

En primer lugar, se han realizado algunas pruebas individuales que han servido para comprobar el funcionamiento individual de cada componente y así poder empezar el proyecto con la puesta a punto correcta de cada uno.

Para el caso de los módulos *bluetooth* HC-06 se realizaron unas configuraciones previas las cuales permiten realizar modificaciones como pueden ser sus nombres o sus códigos PIN de conexión. Estas modificaciones se realizan gracias a los comandos AT que se inician siempre con estas letras (Ejemplos utilizados en la Tabla 4-1) y para su envío se necesita un puerto serie, por lo que se ha hecho uso de la placa de desarrollo de Arduino para pasar correctamente los comandos con las modificaciones pertinentes. Los comandos utilizados para este propósito han sido los que aparecen en la siguiente tabla 4-1:

COMANDO (FUNCIÓN)	TAPA 1	TAPA 2
AT (Comprueba la conexión)	Conectado correctamente	Conectado correctamente
AT+NAMETapa nº (Cambio de nombre)	Cambia de HC-06 a Tapa 1	Cambia de HC-06 a Tapa 2
AT+PIN0000 (Cambio de pin inicial)	Cambia pin de 1234 a 0000	Cambia pin de 1234 a 0000

Tabla 4-1: Pruebas de conexión y modificaciones en los módulos bluetooth.

Para comprobar el estado de las pantallas se ha realizado una programación simple que escribe por pantalla el texto "*Hello World*" que ha servido como referencia para probar que efectivamente están en buen estado y que los pines digitales de la placa de desarrollo A4 y A5 (correspondientes a los pines SDA y SCL en las pantallas OLED) envían bien la información.

Una vez comprobado que los distintos componentes presentan un funcionamiento y estado adecuado, se ha procedido a la elaboración del código presentado en el punto 3.1.2. En la

elaboración de este código se han ido realizando pruebas periódicas que han ido comprobando la funcionalidad con los elementos conectados entre sí, para asegurar que las líneas programadas son reconocidas por los elementos que intervienen en Arduino. Las pruebas que se han hecho a parte de la funcionalidad del código con todos los dispositivos en funcionamiento han servido también para realizar pruebas correspondientes al diseño en la pantalla, como en el caso de establecer un tamaño de fuente considerado suficiente para los casos del contador y la fecha actual por pantalla.

Una vez finalizadas las pruebas de *software* y *hardware* de Arduino, en las que se ha comprobado que ha funciona correctamente el contador y se representa correctamente por pantalla al igual que los datos de la fecha, se ha dado por finalizada la etapa de desarrollo de Arduino.

4.2 Pruebas realizadas en la etapa de desarrollo de la aplicación Android

Al igual que en la etapa anterior, en el proceso de desarrollo de la aplicación Android (presentada en el punto 3.2) se han ido realizando varias pruebas con el propósito de establecer correctamente las funcionalidades que se han marcado como objetivo en esta etapa y observar su funcionamiento en conjunto con lo desarrollado en la anterior etapa de Arduino.

En primer lugar, las pruebas realizadas en esta etapa han sido las de comprobar y habituarse al funcionamiento del entorno de desarrollo y estudiar los distintos elementos que se establecen en la interfaz, junto con sus funcionalidades como es el caso de sensores u otro tipo de conexiones. Todo esto desde la parte de diseño en la aplicación web. En lo que corresponde a la parte de bloques se han realizado algunas pruebas de funcionamiento con respecto al establecimiento de funcionalidades como es el caso de los botones, ayudándose de los elementos preestablecidos de control y lógicos como son los bucles u operaciones lógicas.

Una vez comprobado y estudiado el entorno global, la primera fase de esta etapa se ha basado en el diseño de la interfaz de usuario que cuyo resultado ya hemos visto en el punto anterior. En este diseño se han ido haciendo pruebas con el teléfono móvil personal para comprobar su diseño en un dispositivo real. Ya sea para comprobar la disposición de los botones, textos y configuraciones de *layout* añadidas. El funcionamiento de los botones en este punto era nulo, ya que en este momento no se han establecido todavía las funciones que condicionan el funcionamiento de estos. En cuanto a los textos incluidos al ser meramente informativos de la funcionalidad de cada uno de los elementos introducidos, no se ha hecho más que comprobar que efectivamente se escriben bien y en el tamaño correcto y deseado. En el caso de las configuraciones de *layout*, se ha probado que a la hora de utilizar la aplicación se realizan las funciones y se establecen los elementos de manera correcta. Esto es, que se realiza de manera correcta la función de *scroll* horizontal, que permite arrastrar en la pantalla los elementos que queden fuera de la visión de esta ya que dependiendo del tamaño de cada dispositivo móvil podrá albergar más o menos elementos. Estas han sido las pruebas globales que se han realizado en cuanto al diseño de la aplicación.

Para comprobar la funcionalidad de los botones y otros elementos que forman parte de la aplicación, se han ido probando poco a poco cada bloque o bloques asociados a cada uno

de ellos. Por ello, una vez se establecía la funcionalidad completa en cada elemento que conlleva una acción, se descarga la aplicación para poder comprobar que ha sido programado de manera correcta. Por ejemplo, una vez programado el primer botón de conexión con las tapas, la prueba que se ha realizado ha sido la de descargar la aplicación con la funcionalidad única en este momento, establecida como se ha explicado en el punto anterior en dos bloques, para comprobar que se ha realizado correctamente la conexión con el módulo *bluetooth* correspondiente. Posteriormente se ha realizado el mismo procedimiento de pruebas para cada una de las funcionalidades con lo que se ha intentado asegurar arrastrar los menos errores posibles.

Otra prueba realizada ha consistido en conectar y sincronizar dos prototipos de tapa al mismo *smartphone,* con el objetivo de que este actúe como controlador para varias tapas en función de las necesidades de cada usuario. En esta prueba se ha ido comprobando que las tapas según el caso tienen una buena sincronización si necesitaran llevar el mismo tiempo en ambas, al igual que pueden ser completamente independientes si así fuera necesario.

A la hora de realizar estas pruebas se han tenido en cuenta los objetivos establecidos, sobre los que se ha trabajado como puede ser el caso de intentar ofrecer una interfaz sencilla que tenga un uso rápido e intuitivo.

Una vez finalizadas las pruebas en esta etapa se ha dado por finalizado el desarrollo de la aplicación Android que al estar directamente relacionada con el desarrollo en Arduino se ha pasado a la tercera etapa de integración final.

4.3 Pruebas realizadas en la etapa de integración de elementos finales.

En la etapa final, como se ha explicado anteriormente, se han ido añadiendo y aplicando las mejoras que han completado el dispositivo estableciendo los objetivos restantes que nos hemos propuesto para este TFG. Para este último apartado correspondiente a la última etapa del punto anterior, por falta de tiempo y medios no se han conseguido realizar algunas pruebas más que se tenían pensado como fabricar por medio de una impresora 3D moldes que fuesen capaces de albergar todos los componentes utilizados o hacer uso y pruebas de más baterías que pudiesen tener un rendimiento parecido al que ofrecen las *powerbanks* usadas, pero con menores tamaños y una mayor autonomía. A continuación, se explican las pruebas finales realizadas.

Por un lado, las pruebas realizadas en esta ultima fase han ido encaminadas a comprobar que la soldadura aplicada a los componentes se ha realizado de manera correcta y que las vías de enrutado de los componentes no han sufrido en exceso. También se ha intentado establecer longitudes de cable entre las uniones de los pines aceptables para intentar disminuir el tamaño y aprovechar lo máximo posible el espacio del dispositivo.

Como se ha visto en el punto anterior, las fuentes de alimentación utilizadas han sido dos *powerbanks* de capacidades de 10000mah y 5000mah respectivamente. Se han realizado pruebas solo con estas dos baterías ya que fueron previamente adquiridas para un uso personal y aprovechando que ofrecen un funcionamiento adecuado para su uso que permita al prototipo ser un dispositivo portátil, se ha decidido usar este tipo de baterías como prueba de alimentación. Lejos de ser esto lo más idóneo y con más tiempo y medios, como

se expondrá en el siguiente punto en el apartado de trabajos futuros, uno de los principales objetivos a tener en cuenta en el futuro será el de realizar pruebas con más tipos de baterías. En el caso de las fuentes de alimentación se han hecho dos pruebas con el objetivo de comprobar la conexión y la autonomía del dispositivo global. En la siguiente tabla 4-2 se pueden observar los siguientes resultados de las pruebas con las baterías:

	Batería de 10000mah	Batería de 5000mah
Conexiones	Módulo TTL a USB	Módulo TTL a USB
Autonomía	3 días	1 día y medio

Tabla 4-2: Pruebas realizadas con las *powerbanks*.

Con esto último se han dado por finalizadas las pruebas necesarias para el funcionamiento del prototipo, con lo que finalmente hemos conseguido un dispositivo autónomo y con unas conexiones definitivas adecuadas para todos los componentes que han intervenido en el proyecto.

En la siguiente figura (4-1) se mostrará una imagen real del funcionamiento con la conexión ya realizada con el smartphone del prototipo.

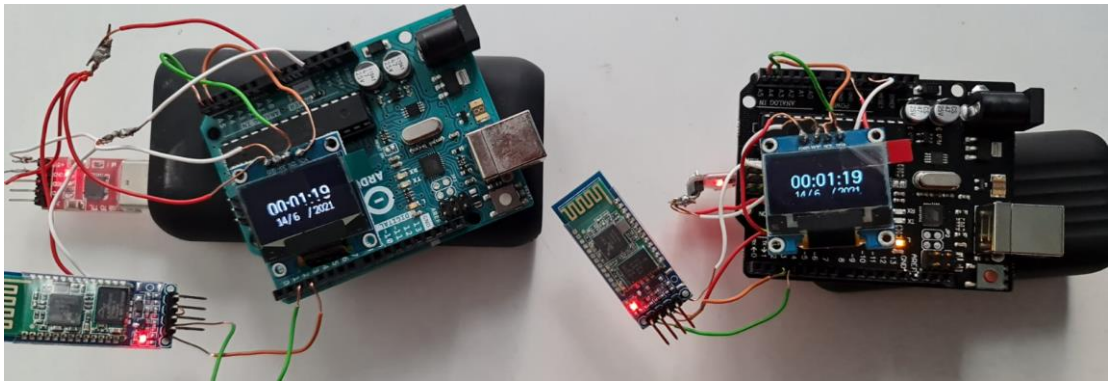


Figura 4-1: Prueba de los prototipos sincronizados en funcionamiento.

5 Conclusiones y trabajo futuro

5.1 Conclusiones

En este Trabajo de Fin de Grado se han construido y diseñado dos prototipos, que pueden ser añadidos a diferentes tipos de tapas de medicamentos, que ayudan en el recordatorio del tiempo de las tomas. Para lograr este objetivo se han investigado y estudiado distintas herramientas y entornos que podían ser capaces de ayudar a conseguir las metas a alcanzar y una vez adaptado a los distintos entornos, se ha procedido a la elaboración de estos prototipos. También se han comentado los elementos que han intervenido en el desarrollo de cada dispositivo como los componentes digitales que lo componen u otras herramientas que se han usado para la unificación de estos. Durante la elaboración del proyecto se han ido realizando pruebas en cada etapa para comprobar que su funcionamiento era el esperado tanto en utilidad como en diseño. Una vez se han obtenido los últimos resultados se han sacado las siguientes conclusiones.

Se han visto las opciones que ofrece el mercado que pueden llegar a tener una funcionalidad similar a la que se pretende ofrecer en los dispositivos desarrollados para este Trabajo de Fin de Grado. Tener constancia de ciertas funcionalidades, ayudó a establecer qué objetivos se iban a perseguir para nuestro prototipo que pudieran mejorar y ser más atractivos para este mercado. Una vez vistos los resultados se ha obtenido, un producto que aparte de que la funcionalidad puede guardar cierto parecido a la mayoría de los productos actuales, es capaz de dar más información que el resto de las opciones ya que se aprovecha también del uso de nuestros *smartphones*. Hoy en día el uso de los *smartphones* está tan extendido que hasta los más mayores pueden ser capaces de poder usarlos y acceder a ellos sin problemas. Por ello una de las primeras conclusiones es que la inclusión de estos dispositivos móviles personales añade más funcionalidades e informaciones relevantes en el mismo producto, mientras que el resto de las opciones del mercado pueden ofrecer las mismas funcionalidades pero en distintos productos, lo que puede hacerlos algo incompletos. Esta conexión entre móvil y tapa a parte de ser muy beneficiosa, es algo que no se ha desarrollado para este tipo de mercado, lo que lo hace algo novedoso y atractivo.

Otra de las conclusiones que se sacan es que no se necesita estar en todo momento cerca la tapa al *smartphone*, ya que la conexión *bluetooth* permite que la conexión con las tapas tenga un rango efectivo de unos 10 metros. Esto permite que al estar en casa su uso sea cómodo ya que normalmente no habrá problemas de conexión y al usar la alarma interna del mismo teléfono móvil facilitará que la persona que necesite realizar la toma reciba al instante y de forma clara el aviso de realización de la toma, ya que el teléfono móvil si suele estar en todo momento con su usuario.

El uso de la tapa en líneas generales es muy simple ya que los prototipos estarán siempre en funcionamiento sin necesidad de realizar ninguna configuración previa ni de pulsar ningún botón, excepto el que tiene de encendido/apagado la fuente de alimentación. Esto unido a la interfaz que se ha intentado programar para un uso simple, hace que el público al que esté dirigido sea de todas las edades y no necesiten de la ayuda de personas con conocimientos más especiales a la hora de usar nuevas tecnologías.

En caso de tener que necesitar más de un bote con medicamentos o pastillero, se ha concluido que la funcionalidad que permite establecer sincronizaciones desde un smartphone a varias tapas es muy beneficiosa. En este sentido tener la posibilidad de llevar el mismo contador en varias tapas o de manera individual, ofrece más posibilidades y ayudas para las personas que requieran de su uso en un sentido u otro.

En general estas han sido las conclusiones que he ido sacando en la elaboración de este Trabajo de Fin de Grado. Como conclusión final, creo que en líneas generales y obviando que es un dispositivo que todavía puede recibir muchas mejoras, de las cuales varias se propondrán en el siguiente punto de trabajo futuro, se ha conseguido desarrollar y diseñar un dispositivo aplicable a cualquier tapa de pastilleros o a botes individuales de medicamentos, con un buen funcionamiento y que realiza el trabajo que se espera de el de manera correcta, con la capacidad de ayudar a personas que tengan dificultades a la hora de depender de si mismos en la toma de medicamentos.

5.2 Trabajo futuro

Una vez sacadas varias conclusiones en la realización de este Trabajo de Fin de Grado, se han pensado varias ideas con el objetivo de tratar de mejorar tanto el uso como las prestaciones del primer prototipo que hemos desarrollado como de la aplicación móvil. Las líneas de desarrollo y diseño a seguir en el futuro son:

- Realización de más pruebas con relación a la fuente de alimentación, que de una autonomía al prototipo mejor que la actual, teniendo como principales objetivos minimizar el tamaño para poder adecuar el prototipo a botes de diferentes tamaños de uso de medicamentos individuales o pastilleros y aumentar la autonomía del proyecto.
- Extender el uso de estos prototipos a otros sistemas operativos como puede ser IOS, perteneciente a dispositivos de Apple y así evitar una exclusividad en Android y extenderlo a todo el mercado sin ningún tipo de limitaciones.
- Mejorar en la medida de lo posible la aplicación, añadiendo nuevas funcionalidades que puedan ser útiles y probar nuevos diseños que pudieran en la medida de lo posible mejorar la interfaz de la aplicación móvil para hacerla más intuitiva.
- Aunque no ha dado tiempo a introducir placas de desarrollo distintas a las Arduino Uno R3 utilizadas en este proyecto si se han empezado a realizar pruebas, con placas de desarrollo hardware en Arduino más pequeñas.
- Una de las principales ideas es la de adaptar la aplicación desarrollada para poder tener la conexión en un mismo dispositivo a más de dos tapas, esto es que desde un mismo *smartphone* podamos controlar más tapas de las programadas en un principio para este proyecto.

Referencias

[1] Timer Cap Company, LLC, “Product Sheet”, https://www.timercap.com/coalitions-drug-misuse-prevention

[2] Timer Cap Company, LLC, “Container Cap With a Timer”, May 21, 2013, Patent No.: US 8,446,799 B2

[3] Timer Cap Company, LLC, “Systems and Methods for Parsing Prescription Information for a Wirelessly Programmable Prescription Bottle Cap”, May 28, 2013, Patent No.: US 8.448,873 B2

[4] E-pill Company, “Instructions for e-pill® Multi-Alarm TimeCap”, 2013. https://sep.yimg.com/ty/cdn/epill/timecapmanual.pdf

[5] E-pill Company, “Automatic Pill Reminder and Dispensing System Program and User Instructions”, 2006. https://sep.yimg.com/ty/cdn/epill/medtimemanual.pdf?t=1621887263&

[6] Rubén Velasco, “Diseña y prueba tus circuitos electrónicos con estos programas”, 2021.

[7] Jörg H. Kloss, “Android Apps with App Inventor: The Fast and Easy Way to Build Android Apps”, 2012.

[8] Juan A. Villalpando, “Cómo programar los teléfonos móviles con Android mediante App inventor 2”, 2021, http://kio4.com/appinventor/19C_activystarter_configuracion.htm

[9] Juan A. Villalpando, “Bluetooth HC-06. Arduino. Send. Receive. Send text file. Image”, May 2020. https://community.appinventor.mit.edu/t/bluetooth-hc-06-arduino-send-receive-send-text-file-image/9518

[10] Emmanuel Zambrano Quitián, “Diseño de una aplicación móvil que permita la gestión oportuna de dispositivos biomédicos en el laboratorio de simulación clínica”, Universidad del Rosario, Bogotá D.C, 2020, pp. 28-30.

[11] Xukyo, “Comunicación con Arduino y el módulo HC-06”, October 2018, https://www.aranacorp.com/es/comunicacion-con-arduino-y-el-modulo-hc-06/

[12] Tsla, “Arduino I2C OLED screen tutorial using U8Glib”, February 2019, https://eecs.blog/arduino-i2c-oled-screen-tutorial-using-u8glib/

[13] Archit Borkar, “Bluetooth Smart Watch Arduino”, January 2021, https://www.hackster.io/architborkar9/bluetooth-smart-watch-arduino-afa813

[14] Saddam, “Build a Smart Watch by Interfacing OLED Display with Android Phone using Arduino”, December 2018, https://circuitdigest.com/microcontroller-projects/build-an-arduino-smart-watch-by-interfacing-oled-display-with-android-phone

[15] Taifun, “App Inventor Tutorials and Advanced Examples”, 2010, https://puravidaapps.com/tutorials.php

Glosario

IDE	Integrated Development Environment
OS	Operating System
MIT	Massachusetts Institute of Technology
OLED	Organic Light-Emitting Diode
TFT	Thin Film Transistor
LCD	Liquid-Crystal Display
TTL	Transistor-Transistor Logic
USB	Universal Serial Bus
TX	Transmitter
RX	Receiver
I/O	Input/Output
SCL	System Clock
SDA	System Data
PIN	Personal Identification Number

Anexos

A Manual de usuario

Este manual pretende ofrecer los pasos a seguir para que, de una manera ordenada y clara, todo el mundo que haga uso de este producto, pueda configurarlo de manera fácil y orientado a cualquier edad.

A continuación, se irán explicando las funcionalidades de la aplicación que conecta con los dispositivos contadores, para aportar al usuario la información necesaria para un correcto uso y funcionamiento de estos prototipos. Primero mostraremos la interfaz global y luego se explicarán las funciones correspondientes a cada botón.

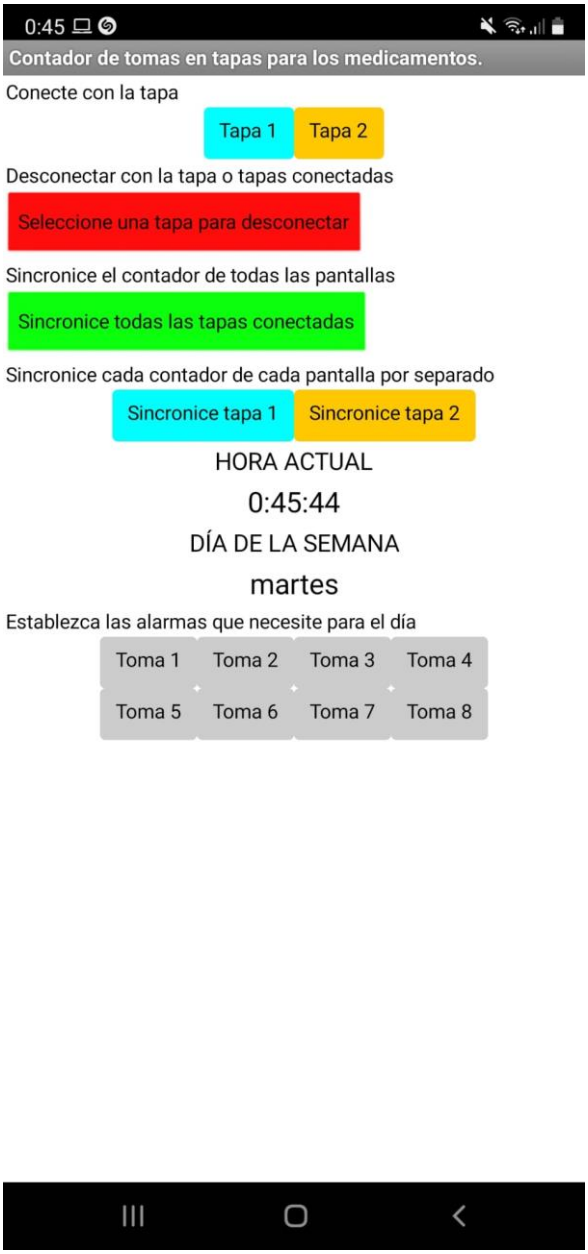


Figura A-1: Vista global de la aplicación en un smartphone real.

1) Conexión con las tapas:

Figura A-2: Botones de conexión con las tapas.

Los botones que aparecen en la imagen serán los que permitan conectarnos a las tapas que tengamos disponibles. Al pulsar uno de estos botones, aparecerá una lista de dispositivos bluetooth disponibles entre los que tendremos que seleccionar el nombre que tenga asignado cada producto.

Una vez elegido el dispositivo conectado, saltará en la pantalla de nuestro dispositivo móvil un aviso que indica que se ha realizado la conexión de manera correcta con el dispositivo o en su defecto saldrá un mensaje de error. También se puede comprobar que la conexión se ha realizado correctamente con la tapa comprobando que el LED parpadeante perteneciente al módulo bluetooth ha pasado de estar parpadeando a estar en encendido continuo.

2) Desconexión con las tapas:

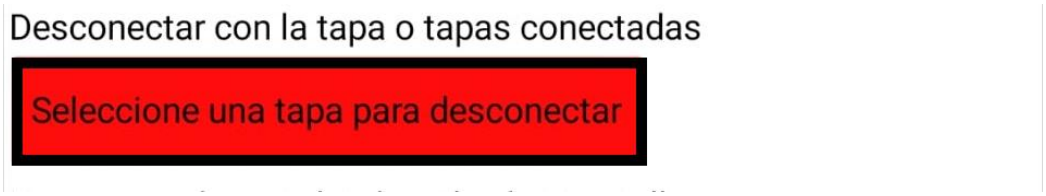


Figura A-3: Botón de desconexión de las tapas.

En el caso que se necesite desconectar de una tapa, el botón en rojo permitirá seleccionar de la lista de dispositivos bluetooth, aquellas tapas que se encuentren conectadas a nuestro móvil.

Una vez seleccionado el dispositivo del que deseemos desconectarnos, saldrá un mensaje de confirmación indicando que el dispositivo se ha desconectado correctamente. En caso de no encontrar dicho dispositivo, saldrá un mensaje de error indicando que no se ha encontrado el dispositivo a desconectar.

3) Sincronización del tiempo global de las tapas:

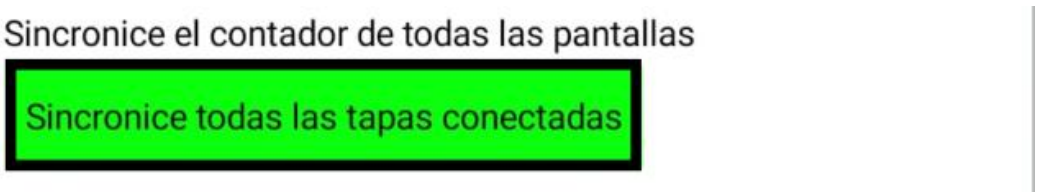


Figura A-4: Botón de sincronización global de las tapas.

En el caso del botón verde, se establece una funcionalidad, mediante la cual, si el usuario necesitase que el tiempo transcurrido desde la última toma tuviera que ser el mismo en todas las tapas a su disosición, este botón se encargaría de sincronizarlas.

Una vez se pulsa, el tiempo en todas las tapas empezará a contar desde cero, estableciendo el mismo tiempo transcurrido en los contadores conectados.

4) Sincronización del tiempo global de cada tapa:

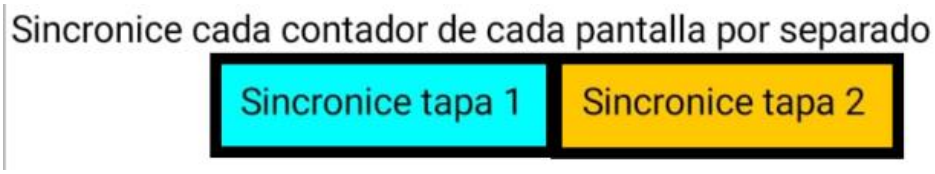


Figura A-5: Botón de sincronización individual de cada tapa.

Para los dos botones representados en los mismos colores que en los de conexión con las tapas, se ha establecido una funcionalidad parecida a la vista para la sincronización global de las tapas, pero en este caso se realiza de manera individual para cada tapa.

Si el usuario realizase tomas distintas para cada tapa y necesitase llevar un tiempo de conteo distinto en cada una, podrá reiniciar el contador de cada una por separado con ambos botones.

5) Información de hora y días actuales:

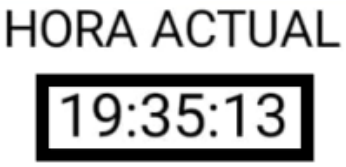


Figura A-6: Información de la hora actual.

Se establece un reloj con la hora actual, para facilitar al usuario el tiempo actual y que puede relacionar con el tiempo transcurrido con el contador progresivo para tener toda la información posible de cara a la siguiente toma.

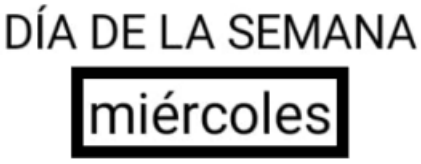


Figura A-7: Información del día actual.

Otra información que se considera interesante saber es la del día actual, ya que en caso de ser tomas semanales, es importante saber el día de la semana actual para que en el caso por ejemplo de los pastilleros semanales se tenga toda la información posible al respecto.

6) Programación de las alarmas:

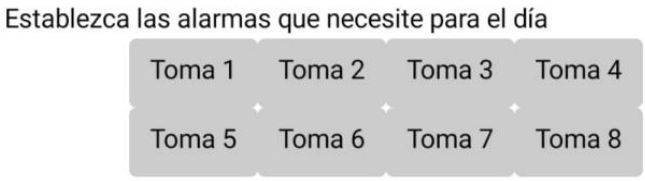


Figura A-8: Botones de programación de las alarmas.

Para programar las alarmas, se dispone de ocho botones de selección horaria en los cuales se establecerá la hora en la que se necesite realizar una toma.

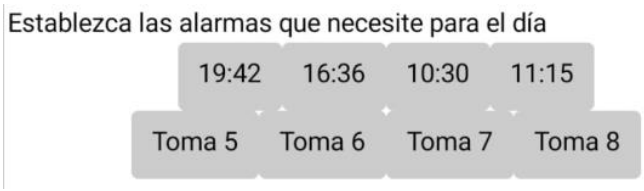


Figura A-9: Botones de programación de las alarmas con ejemplos programados.

Como se ve en el ejemplo, se han programado un total de cuatro alarmas en las cuales aparecen los tiempos en los que van a alertar las tomas. No es necesario programar las ocho tomas ni establecerlas en un orden determinado.

Si se quiere quitar el tiempo representado en un botón, bastará con volver a pulsar el botón y cancelar la selección horaria. Para quitar la alarma programada nos iremos a la parte del reloj del teléfono donde están las alarmas y la deshabilitaremos.